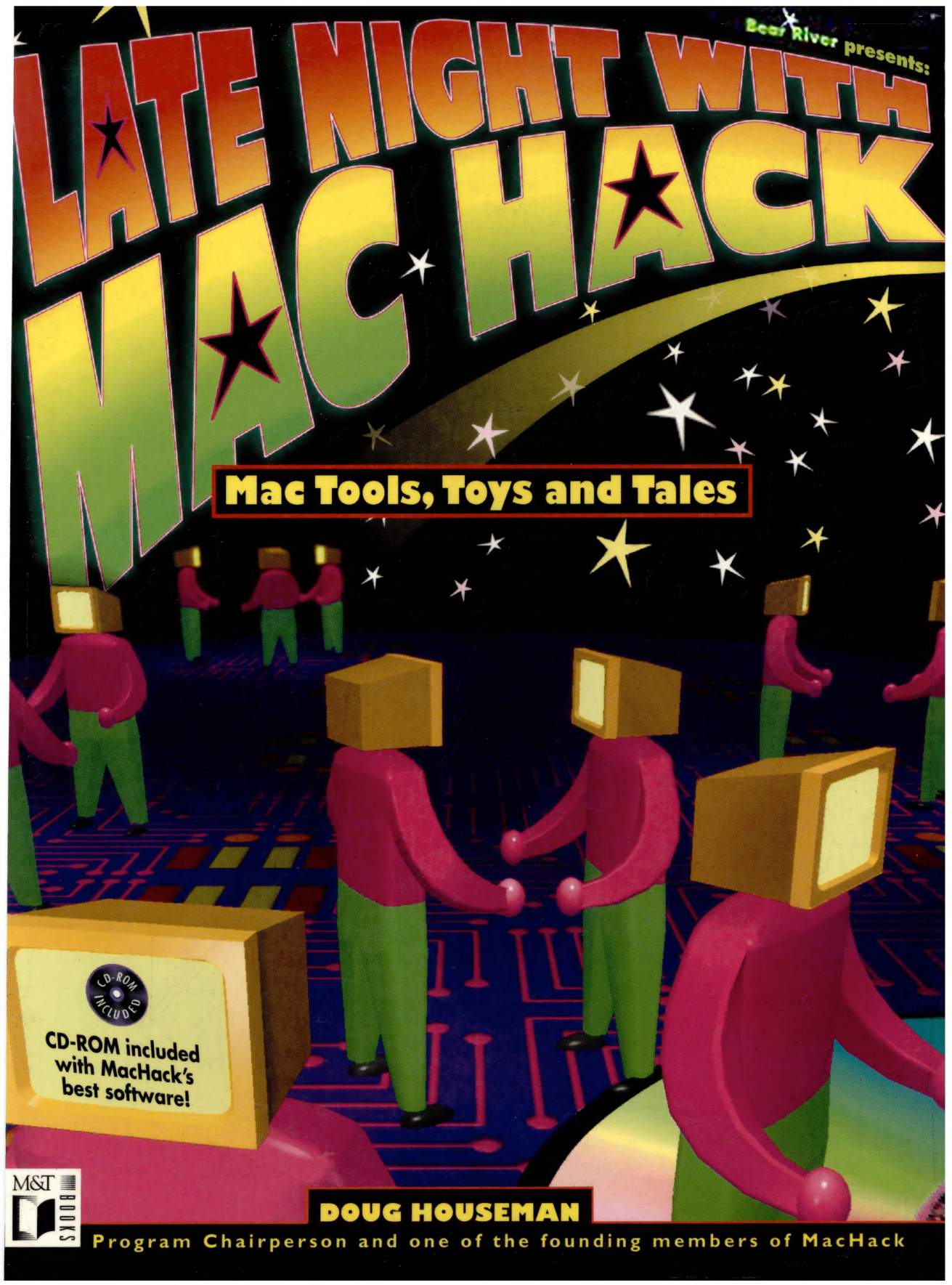
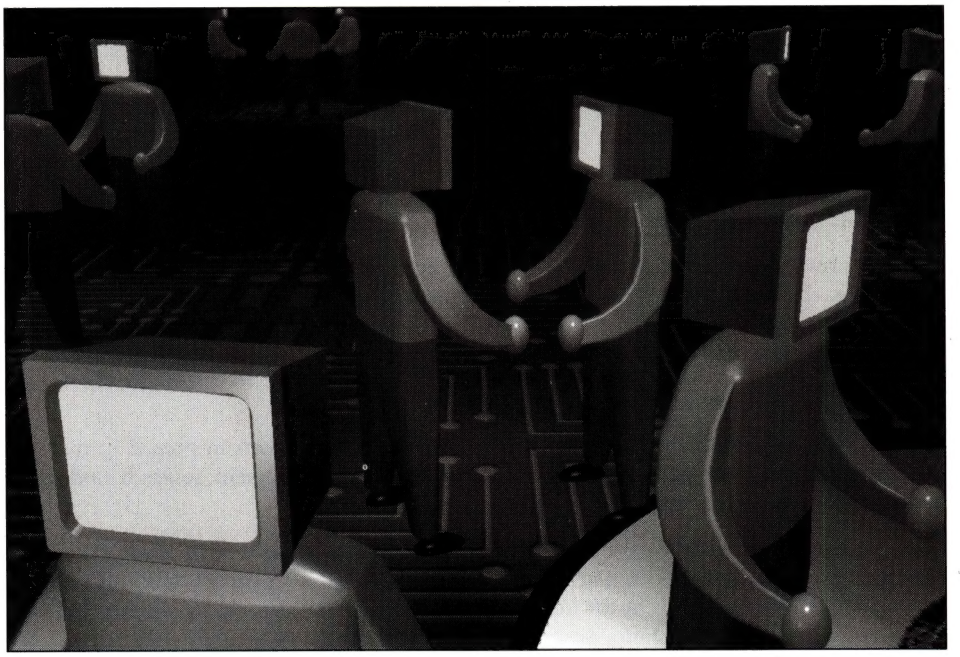

Bear River presents:
LATE NIGHT WITH MAC HACK
Mac Tools, Toys and Tales
CD-ROM INCLUDED
CD-ROM included with MacHack's best software!
M&T BOOKS
DOUG HOUSEMAN
Program Chairperson and one of the founding members of MacHack



Late Night with MacHack

MAC TOOLS, TOYS & TALES

M&T BOOKS

M&T Books
A Division of MIS:Press
A Subsidiary of Henry Holt and Company, Inc.
115 West 18th Street
New York, New York 10011

Limits of Liability and Disclaimer of Warranty

The Author and Publisher of this book have used their best efforts in preparing the book and the programs contained in it. These efforts include the development, research, and testing of the theories and programs to determine their effectiveness.

The Author and Publisher make no warranty of any kind, expressed or implied, with regard to these programs or the documentation contained in this book. The Author and Publisher shall not be liable in any event for incidental or consequential damages in connection with, or arising out of, the furnishing, performance, or use of these programs.

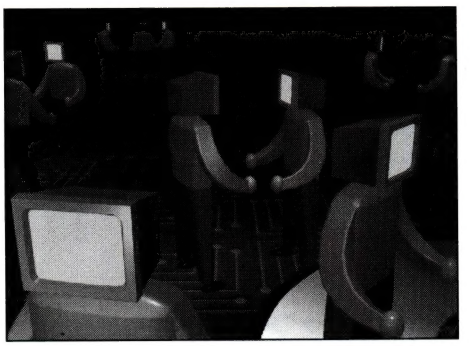
Hacks use with permission of Expotech.

Library of Congress Cataloging-in-Publication Data (to be assigned)

Houseman, Doug
Late Night with MacHack: Mac Tools, Toys & Tales/Doug Houseman
p. cm.
Includes Index
ISBN 1-55851-395-7

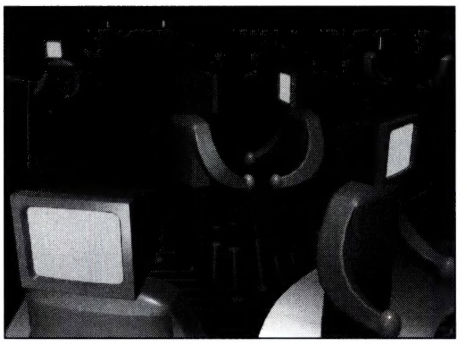
96 95 94 93 4 3 2 1

Publisher: Brenda McLaughlin
Development Editor: Laura Lewin
Production Editor: Anne Alessi
Associate Production Editor: Erika Putre
Copy Editor: Suzanne Ingrao
Technical Editor: Joel Nagy



Dedication

This book is dedicated to Ann Marie, without whom the book would not have been possible. Thanks for all your love and brownies.



Acknowledgments

My thanks to: Amiée, Bobbie, Carol, and Heather, without whom there would be no MacHack; Scott and Greg for creating the Hack contest; Laura Lewin for being patient when the deadlines were missed and for the words of encouragment; Joel Nagy my sounding board on the book. Raines Cohen for last minute help in the final all-nighter. Thanks for returning the favor; and to all the hackers who have made the Mac such a wonderful fun machine. Please keep it up!

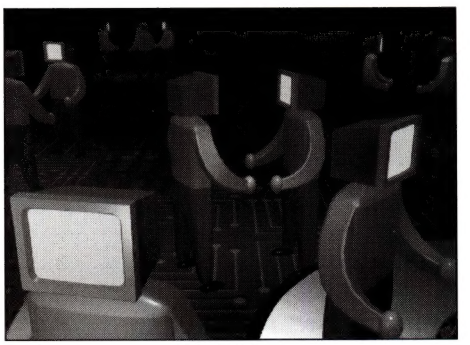
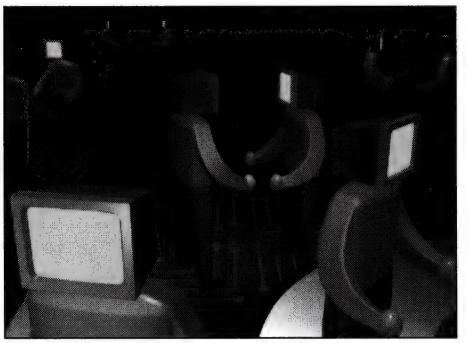


Table of Contents



Introduction

In a small midwestern university town known for its football team, there is an annual conference that has had more impact on the computer industry than any other. This conference has brought together a group of programmers and industry gurus since 1986. It started as an educational conference. It has mutated into a conference that only a programmer could love. The conference starts at midnight and runs around the clock. Two computer labs, complete with networks and internet connectivity, are installed in one day. Seventy-five machines silently await the talented fingers of 300 of the best programmers from around the world. At stake is a prize coveted by all: the *Best Hack Award*. It means more than any other prize, award, or trophy that is given in the computer industry, and not because it will sell more product, or give you more money. It means more because it is the only peer award.

The conference is MacHack. Since 1986, over 1000 programmers have attended. They have created a family in the Macintosh industry that has allowed innovation to quickly spread. A family that allows a single programmer to be involved on a dozen or more best selling products from four or five companies. It has been the creation point of many innovations. Desktop printers, multicolor icons, and inter-application communications were all created at MacHack. Net Bunny,

Wavy and DOS sHell all saw the light of day for the first time at MacHack. Over 300 little programs, called *hacks* were created for MacHack.

MacHack is also about people, young programmers with ideas and talent, getting a chance to show off to the old guard. This has lead to the hiring of new super star programmers by a number of companies. Apple has MacHack to thank for the software integration team for System 7, the *Blue Meanies.* Of the 13 members, eight of the *Blue Meanies* were hired because of MacHack. Apple's QuickdrawGX, HyperCard 2.0, AOCE, the Newton, and more have benefited from programmers discovered at MacHack. Other companies have benefited, as well. Microsoft, Now, 5th Generation and more have found key people or products at MacHack.

MacHack is about the future. Of the hacks that were created at MacHack, over 50 have gone on to be part of system software or a commercial product. Watching four programmers sharing a machine and an idea can be a lesson in creativity. Each programmer puts his or her own spin or signature on the hack that they are creating. As more programmers become involved, the hack takes on a life of its own. In each case, the participants walk away with new knowledge and ideas. Those ideas become the foundation for the next generation of Macintosh technology.

MacHack is about sharing. Programmers stand in front of their peers and offer information about the projects that they are working on, how a particular piece of code works, how to make a printer run, or how a piece of technology could be used to completely change the way computers are used. From wireless networks to accelerator boards, the technology is tested on overhead projectors at MacHack, before it goes on sale. MacHack is rough on products that are not well thought out.

MacHack is about fun. The dinners and the movies at midnight. Programs like Wavy and NetBunny. Jolt and cold pizza at 3 AM. The Hack contest at 2 AM. Changing the sign at a movie theater or the hotel. Harmless fun that gets everyone rolling on the floor. Laughter as the keynote speaker is covered in silly string by his friends, or when

a boring speaker is bombarded by nerf balls. Peers testing peers and making friends.

MacHack has become an institution. The local Jolt bottler donates at pallet of Jolt, the Pepsi bottler a pallet of Mountain Dew. Domino's and Cottage Inn Pizza both schedule extended hours, and the theater reserves a midnight showing. The hotel staff has gotten used to keeping the lights on in the *Holidome* all night long and shifting the maid schedules to late afternoon.

Author's Note

I have been privileged to be part of the MacHack family almost from the beginning. I was a graduate student at the University of Michigan in 1986 and was invited to represent our user group at the conference. When I arrived, people needed banners printed, so I went home for my Mac and Imagewriter. I became part of the staff, because I was willing to create signs and banners. That afternoon, I was asked by several people, for a copy of the banner program. The next day, I brought the whole user group library to the event. Needless to say, I did not see any of the sessions that day. I was asked to assist with the 1987 conference and ended up as the program committee chair. What I did not know at the time was that the program committee was just me. MacHack was a success in 1987, and we were off on a roll. The committee has grown and changed over time, but the focus has not. It is the search for the best information from the best sources. For example, the lead programmer on a project is usually our main speaker, rather than the marketing person. Marketing people have their place (I should know, I am one), but it is not at MacHack. Another thing, MacHack is not just for programmers. We have a number of technical writers, press people, venture capitalists, and others who have joined our band. There is an interesting statistic about MacHack, and that is that, if you have been to two, you will most likely come to the rest. For what other conference holds the keynote session one minute after midnight? MacHack knows that most of its participants will be awake and fully functioning then.

What is MacHack?

MacHack is a technical conference that brings together 300 (by the way, attendance is capped) of the top Macintosh programmers from around the world for 4 days to test their skills in friendly competition. It is held in Ann Arbor, Michigan, which is also the home of the Michigan Wolverine football team and the Fab Five basketball team. It is held every June and is usually interrupted by at least one thunderstorm. It is sponsored in part by the University of Michigan, APS, Microsoft, Apple Computer, and other companies in the industry.

The conference kicks off at midnight on the opening day, with a review of the Hack contest rules, what hardware is in the machine room, and a welcoming speech from the volunteer chairperson of the conference. The technical track continues until 4 AM, when a halt is called for the night. The day picks up at 8 AM with the opening of the business track, including sessions on how to run your own software or manage a consulting firm. By 10 AM, the papers track is rolling, with papers on topics like Multiprocessing Machines, and Electronic Conferencing as a method of software development. By Noon, the technical tracks pick up. Brunch is at 1 PM and then the day really starts to roll. Powerbooks are plugged into the network, and the 75 machines in the machine rooms are all taken. By 4 PM, the machine room is full. By 6 PM, there are no more connectors on the wireless network. By 10 PM, the Holidome is full of programmers, quietly creating software. Midnight comes and goes, Domino's makes a last call for pizzas at 3 AM, then Cottage Inn at 4 AM. The technical track ends at 4 AM or so, when the day once again starts anew. Many feel that, if you sleep, you are not a true hacker.

Each year, the number of non-disclosures required to attend all of the techincal sessions has grown. Many topics covered at MacHack are also covered at Apple World Wide Developer's Conference. The dif-

ference between the two conferences is that at MacHack, the principle programmer is at the front of the room and there are only 30 or 40 people in the room. At Apple World Wide, there are 2000. The more intimate conditions at MacHack can lead to rather spirited discussion about the technology. More than one Apple technical program has been reshaped at MacHack. The give and take in a room with 40 interested people is much greater than that in a room of 2000. And don't forget that MacHack is not politically correct in the eyes of most marketing people, as it is known to argue healthily with Apple's current position. One company shelved a product because of its reaction at MacHack. Several other products have become best sellers for the same reason. We are an irreverant bunch, but *MacWeek*, *MacUser*, and *MacWorld* all send reporters. It is one of the few small conferences that gets editorial space in all of the major publications.

What is a Hack?

A hack is a short program that allows a programmer to demonstrate one thing. In the case of Color Finder, it was full color icons on the desktop. Hacks are normally written as a demonstration program of a product's possibility, rather than as a product. They are not fully tested and lack quality assurance. In fact, they tend to crash a lot.


WARNING

Is it safe to use hacks? Sure! There is no destructive code in any of the hacks that were created at MacHack. The hacks are unstable, because of lack of testing time, but they are not designed to crash your computer. If you use them, you should first back up the machine, and be prepared for a crash.

Many hacks will run for days and weeks without crashing, but sooner or later, they will crash. The programmers wrote great code, it is just that the hacks were never really tested and crash proofed.

How Can I Tell if a Hack Will Run on My Machine?

In this book, as each hack is described, you will find a set of codes that tell you what the hack is able to run with. These codes are as follows:

6	System 6.02 thru System 6.0.8 compatible
7	System 7.0 thru System 7.1 compatible
~~040~~	not compatible with the 68040 with the cache turned on
Newton	requires a Newton to be used
AE	requires AppleScript extension to be installed in order to run
HW	requires specific hardware in order to run, which is described in the text
SW	requires specific software in order to run, which is described in the text
§	source code for this hack is included on the CD-ROM.
PPC	requires a PowerPC

If you follow this guide and additional information that may be in the text, you should have no problems.

What is a Hacker?

The term *hacker* dates back to the MIT model railroad club. It was a term that was used to refer to people who could really make the railroad run. The term was adopted by the early student programmers who used it to refer to programmers who could really make the computer work for them. With the limited memory of the early computers, very small programs that were very efficient were critical to getting things done. Therefore, hackers wrote hacks to make the computers work best for people. The best hackers wrote the most critical programs. Systems hackers were better than application hackers, and hardware hackers were top of the list. Programmers still use the term hacker to refer to very good programmers. You might have heard hackers referred to as electronic criminals, but this is not true. They are called pirates or crackers among the programming community. And contrary to popular opinion, most crackers are caught by hackers. Or to put in another way, the hackers wear the white hats.

What is on the CD-ROM?

The CD has three main parts, as shown in Figure I.1:

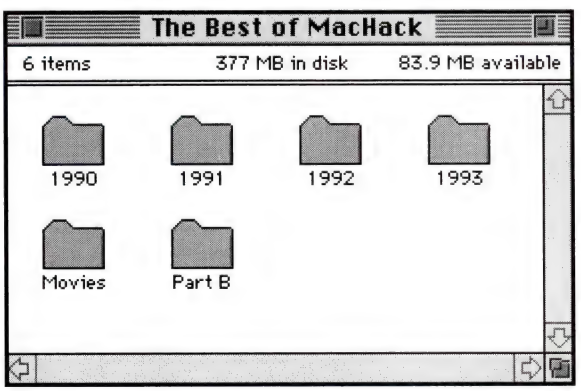


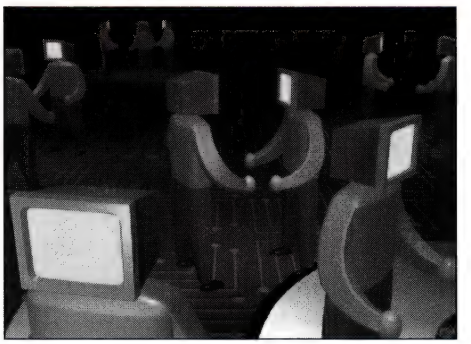
FIGURE I.1 DESKTOP VIEW OF THE BEST OF MACHACK CD

The hacks, are detailed in the book and in *Appendix A*. They are contained in the folders **1990**, **1991**, **1992**, and **1993**.

QuickTime conference proceedings and movies of MacHack as it happened are contained in the folder named **Movies**. Most of these movies date from 1992 and 1993.

You will also find material in the folder named **Part B**, including slides from talks, copies of papers and other miscellaneous material that interests programmers, which is detailed in *Appendix B*.

CHAPTER 1



MacHack—The Early Years

1986—The Beginning

One of the first major institutions to commit to Apple's new computer, the Macintosh, was the University of Michigan (U of M), which was one of the 30 members of the Apple University Consortium. As a consortium school, the U of M received special status and help from Apple Computer. The U of M was expected to use the Macintosh as an integral part of the curriculum for students. While this was fine for a number of computer courses, programs to integrate the Macintosh into the vast majority of courses did not exist. There were word processors for English, but there was no software to support classes in foreign language, or biology or chemistry. Professors, graduate students, programmers, and staff had to create courseware for the Macintosh. While this did not seem like a daunting task in 1984, when the University Consortium was formed, by late 1985, it was obvious to many people at the U of M that courseware was not being created fast enough. Programming for the Mac interface required more new skills than most people had imagined. Additionally, there was difficulty distributing courseware to the students at the 30 colleges that were involved in the program. And the small installed base of the Mac in 1986 did not yet warrant interest by major book publishers and the major software publishers. Apple did not yet have the infrastructure to

market the Mac, and the Universities did not want to build one. The U of M had a very large investment in technology that most of the students were not yet benefiting from. Something had to be done.

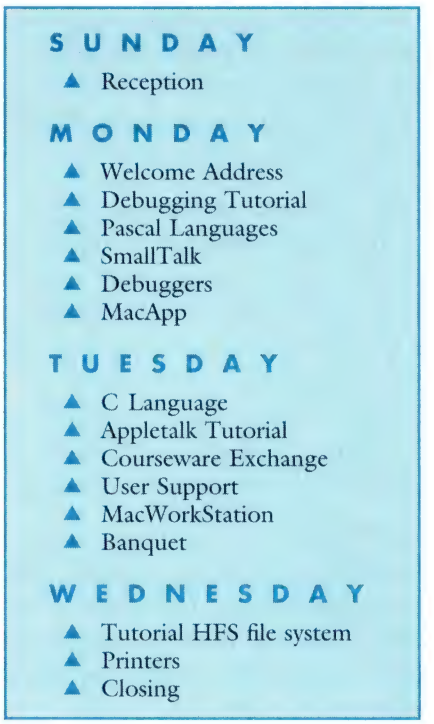
Ann Gordon, a graduate student at the University of Michigan and the driving force behind the U of M Macintosh User Group, met with several professors to discuss a conference about courseware for the Macintosh. This conference would be used to exchange information on how to program the Macintosh, to explain Kinko's courseware distribution system and to explain what courseware and tools were available. Ms. Gordon met with Dr. Gavin Eadie of the Computing Center, Dr. Carl Berger of the School of Education, and Dr. Karl Zinn and John VanRoekel of the Computer Aided Engineering Network (CAEN) to discuss the conference in the fall of 1985. It was decided that the U of M should host the conference with the assistance of Apple Computer and Kinko's Copy Centers. Ms. Gordon contacted Apple's local office in the Detroit area for assistance, and Steve King, a Field Service Engineer was assigned as Apple's representative to the committee and to the conference. The group of five met on several occasions in the late fall and early winter of 1985. They made great progress on the content of the conference and who to invite. The committee came to the conclusion that there was a need for the conference and that there was enough interest, but none of them had ever hosted a meeting like this before. They decided that a professional meeting planner was required to handle the logistics and registration for the conference. In January 1986, with six months to go to the conference, Amieé Moran of Expotech was engaged to handle the logistics of MacHack.

The details of the conference needed to be confirmed. Ann Gordon acted as point person for the university and Steve King as the point person for Apple. Amieé's job was to coordinate all of the publicity, marketing, site selection, and registration. Ms. Gordon was responsible for the program, and Steve was responsible for all the equipment for the conference and Apple's participation. A local hotel was found to host the conference, and material about the conference

was sent to all of the consortium schools. Attempts were made to mail information to the Apple Developer Community, but Apple Corporate felt that the conference was too focused to be of use to the general development community. Marketing the conference was difficult without this list, but the conference was still on.

During the spring, a problem arose: A Macintosh programming company called "The MacHax™ Group" wanted to ensure that there were disclaimers in the program book and elsewhere that the MacHack conference was not affiliated with the MacHax Group. This caused more than one person at the University of Michigan to consider canceling the conference. With diligence, Ms. Gordon and Ms. Moran, managed to keep the conference on track and the MacHax Group out of court. Unfortunately, there were no disclaimers on the printed material and the matter remained unsettled for another year.

As the conference date grew nearer, it became obvious that the conference would not reach the goal of 100 attendees. Fewer than 50 people were preregistered and would attend. The program was limited by the availability of space and equipment to two simultaneus technical sessions that covered all the basics: programming, debugging, tools, marketing, distribution, and case studies. In each area, a specialist was engaged to make the most of the program. The U of M staff covered Pascal in both TML and Think Technologies. Pascal was the favored programming language of the conference (and the Macintosh community then), and many sample code pieces were discussed and traded between attendees. Russ Daniels of Apple Computer covered MacNosy and TMON, the debuggers that were available for the Mac. Larry Rosenstein covered SmallTalk, the up-and-coming development system for the Macintosh. Russ went on to head the project that produced SADE and Virtual User for Apple. Larry has been involved since in many projects at Apple. Martin Haeberle of Apple covered MacApp in its first incarnation, and Invention Software covered the Programmer's Extender Series. The case studies included Ann Arbor Softworks (FullWrite, FullPaint), C as a programming language, and working on a solo project. The full program is listed as follows:



Additionally, there were four case studies on the schedule and a number of unscheduled sessions happened. While the program does not look very full, the topics were covered in depth and the attendees spent up to two hours asking questions after each session. Attendance was light (only 63 people came to the first MacHack). There was no programming or "hacking" at the conference; instead, it was very academic with references made regarding its typical university characteristics. A small vendor's room was available, and Apple, Kinko's, and the local user group MacTechnics were the exhibitors. Absoft and Invention software had tables, but did not staff them because of the light turnout. The favored exhibit was the user group exhibit, because the group had brought a Mac and a printer, which was used to copy disks for the attendees.

The attendees enjoyed the conference, which planted seeds that would grow into Player Piano and finally Virtual User, the automated

test platform for the Macintosh. Later, the same seed started Apple Events and the Apple Open Collaborative Environment (AOCE).

When Ms. Gordon graduated from the U of M in the spring before the conference, they lost their driving force. The conference was also more time-consuming than the University intended, and it lost some money the first year. Therefore, no plans were made for a second conference. A number of attendees asked Amiée Moran if Expotech could organize a 1987 conference. Amiée agreed, under the condition that some of the 1986 attendees had to assist in designing the 1987 program and in marketing the publicity. Hence, the MacHack committee was born.

Macgician—Leonard Rosenthol

Leonard Rosenthol is one of the most quoted Macintosh programmers of all time. He is a frequent contributor to the various Macintosh Usenet newsgroups on the Internet, as well as to discussions on America OnLine and Compuserve. Leonard attended the first MacHack as a student and as a staff member of Davka, a software company that was creating the first desktop Hebrew word processor. Leonard, the technical genius behind the Davka products, spoke out on the need for additional MacHacks and was one of the first three people to sign up to be a MacHack committee member. Leonard left the Chicago area after finishing school and moved to the West coast. He started with a company called Lazer Ware, making printer drivers and other printing related software.

He later became the lead Macintosh Programmer for Software Venture's Microphone family of products. After bringing Microphone through a major overhaul, he accepted a job offer at Aladdin Systems, the makers of StuffIt!. Because his home was in the Berkeley hills and Aladdin Systems was in the South Bay, Leonard decided to move south. Although he was in the Scotts Valley area looking for a new apartment, the Berkeley fire struck, leveling his apartment and all of his personal belongings, he still attended MacHack and provided yet another of his hacks at the conference. Leonard is the driving force behind the desktop printer concept that Apple has built into

QuickdrawGX and Adobe has embraced in Postscript level 2. He has also created several programming libraries and other hacks. AutoStuff, StuffIt! Extractor and Drop Stuff are all utilities written by Leonard. His current product, SitComm!, the new telecommunications program from Aladdin, just won his second Eddy award from *MacUser*. Leonard gave freely of his time and energy to disassemble a number of virus samples and helped John Norstad and others develop the antivirus programs that now keep most of the Macintoshes safe from infection. At last count, over 200 utilities and programs for the Macintosh were either written by Leonard or contained some of his code.

1987—Apple's One-Man Band

The balance of the summer of 1986 came and went. No one was sure if there was going to be a MacHack in 1987. Finally, there was a meeting between some of the people who volunteered to be on the MacHack committee and Expotech. MacHack had generated enough publicity in the various Macintosh publications that many people wanted to know more and still more wanted to attend. It was obvious that the U of M would not take the lead or the financial risk in the next conference. It was also obvious that none of the volunteers had deep enough pockets to cause the conference to happen. Expotech made the decision to go ahead with MacHack in 1987.

The committee was formed with David Feldt of Broadacre Network, Karl Zinn of the University of Michigan, Rod Ganiard of MacTechnics, and Carol Lynn and Amieé Moran of Expotech. These groups pulled together a conference that had over 20 sessions and 30 speakers. The university agreed to continue to sponsor MacHack, under the provision that the vast majority of support would come through equipment donations. This equipment would be used in the Monday and Tuesday sessions and tutorials, which were added for 1987. Apple declined to support the 1987 conference, but the local Apple office, through Steve King, came through with critical equipment when it was needed.

The Marriott, the hotel for 1986, was sold to a new owner and was not available for MacHack. The only larger Ann Arbor hotel was the

Holiday Inn West. Since there were no July dates, the conference had to move into June, and MacHack has since been at the Holiday Inn West in the first half of June. The conference was lengthened to a full week, with tutorials on the Macintosh Programmer's Workshop on Monday and Tuesday. The session topics became more technical, with subjects like "The Variations in the Macintosh ROM" (where most of the operating system used to live) becoming the standard. Object Oriented Programming, Tools, Languages, and Multiuser Applications were the hot topics at MacHack. David Lingwood of APDA and David Smith of *MacTutor* were in attendance, and generated pre- and postconference publicity. In turn, the magazines each signed several regular contributors. Scott Boyd, part of the MacHax Group, signed on with *MacTutor* and is the editor of its successor, *MacTech*. David Lingwood attended the 1986 MacHack, and was the director of APDA (David felt that Apple should be more involved in the future of MacHack and should support the conference). His liaison at Apple was a young manager, who had started as a programmer, Jordan Mattson. Jordan was a high-energy person with a great sense of humor. He was willing to add funds in his budget to come to MacHack, because David asked him to and because he agreed that it was "the right thing to do."

As the countdown to the conference occurred, the committee became a little apprehensive; Scott Boyd of the MacHax group was going to attend. No one could predict the results of this meeting. Scott's letters were very blunt and legalistic, but, on the phone he was very polite and unassuming to a fault. However, it turned out for the best; Scott enjoyed MacHack and joined the 1988 committee. The problems with the name were cleared up, and Expotech registered it as a ServiceMark. MacHack now had its identity.

The tutorials were well attended, and most of the attendees stayed for the conference. The tutorial equipment was retained for evening sessions and for the speakers. This equipment ended up being heavily used and formed the first MacHack machine room. Since development software was loaded on all the tutorial machines, it was easy for a programmer to slip in and test something that they had just discussed during a session. Speakers were frequently taking attendees to the computers for hands-on demonstrations. Leonard Rosenthal was con-

stantly assisting people with code they brought, because they needed help. This evolved into the code clinics that Leonard has run ever since. The machine room has grown into the informal gathering place for the attendees.

Doug Clapp, a columnist for *MacUser* was invited to give the first MacHack keynote speech, which was about MacBots. Doug had written two columns about how the Macintosh looked friendly enough and had all the right parts in it to make a good household robot, indicating that all it needed was to see, hear, move, and talk. Macintalk existed, so that the talking part was easy. This inspired a number of programmers to go out and attempt to write a MacBot operating system, and it also inspired the Banana jr. computer, which would grace the comic pages of the newspapers a few months later. With the advent of the Macintosh AV class machines in 1993, Apple took the first step in a long time toward Doug's goal. There are still a number of hard problems to be solved. None of the 1987 attendees understood what they were attempting. Millions of dollars have been spent since to make pieces of this MacBot real.

The first of the Macintosh II computers were available at MacHack, and the programmers went nuts over them; color on the Macintosh was quite a concept! After all, for three years, the Mac was black and white, and color opened a whole new dimension for the Macintosh interface to explore. Everyone wanted to try their hand in a color quickdraw, the built-in drawing routines on the Mac II. This material was not in the conference schedule, but consumed hours of time in the discussion areas. It became obvious to the committee that topics needed to cover the very edge of the technology as well as the standard programmer issues of tools and debugging.

Jordan Mattson, the only Apple attendee, was everywhere. There were portable walls between the sessions, and Jordan would open the front section so that he could listen simultaneously to two discussions. When an opening occurred for Apple information, Jordan would enter that room and make comments and then step back into the other room. He made quite an impression on some of the attendees! There were a number of discussions about the "people" Apple sent, as if Jordan

had been triplets. Jordan agreed to join the committee in future years and to assist in getting speakers and participation from Apple.

In 1987, Jordan was Apple; he hired a number of the attendees from MacHack for Apple and commented on almost every topic. He was very good at providing information and in taking information back to Apple. The programmers in attendance were angry enough at Apple so that there was a "Bash Apple" session scheduled at the last minute to allow a forum for Jordan to answer questions, clear the air, and receive feedback. The people were so angry, a moderator with a Louisville Slugger was needed to hold the audience at bay; hence, the name Bash Apple. Jordan worked the crowd like a seasoned politician, politely and thoughtfully. By 10 PM, Jordan had the crowd convinced that Apple cared. The session carried on until almost midnight. Jordan was hoarse after the session and talked himself hoarse again both Thursday and Friday. Every question that could not be answered was written down. Jordan stayed up most of the night composing messages and getting straight all the information. Every question was acknowledged and answered by someone at Apple in the following weeks; Jordan was now the MacHack evangelist at Apple.

Amieé was asked to copy a number of disks for attendees with sample source code and some applications that were written at the conference. This set of five floppies was the first real proceedings from MacHack. Only a few copies of the set were produced, and they were all given away that day.

The attendance broke 100 in 1987, and the conference was a mild success. Expotech formed a new volunteer committee to make sure there was a 1988 MacHack. Everyone left the conference happy, and the MacHack family began to form.

Apple corporate decided to support a different MacHack in 1987—MacHack West at the University of Oregon. Steve Wozniak was the keynote speaker. A number of the MacHack attendees were angered by Apple's refusal to support the first MacHack, followed by their choice to support a rival conference. There was a letter campaign to protect the MacHack name. Since Expotech owned the ServiceMark to MacHack, Apple legal became involved. After the exchange of a

number of letters, Apple produced an agreement that allowed MacHack West to occur in 1987. For a few months, everyone awaited the outcome of the MacHack West negotiations. Jordan worked within Apple to answer questions and make peace. As a result, the University of Oregon decided to run just the conference in 1987 and Apple decided to support the 1988 MacHack conference. Four months later, there was Doug Clapp who wrote a column in *MacUser* that put MacHack over the top. The phone did not stop ringing for the next month. MacHack had found an enthusiastic audience.

Darin Adler—from Teenage Game Designer to System 7

A small company in Chicago was pioneering games on the Macintosh. Taking the text adventure one step further, they created a graphics engine that allowed text adventures with graphics. Deja Vu is the first and best known of the games written with this engine. The company was Icom Simulations. There were two young programmers that were the basis of the company: Darin Adler, the designer of the game engine and Waldemar Horwat, the designer of TMON, their debugger. Darin was still in high school when he attended his first MacHack in 1987 as an invited speaker on the languages panel. Darin had more than a working knowledge of both the available pascal compilers for the Macintosh and the C compilers. Despite his youth he easily proved that he was a better programmer. Darin stayed with ICOM for another year, until early 1988 when Apple recruited him to become a programmer in the system software group. Darin tackled programming as if there were no bounds and no rules. He wrote code that broke many programming conventions, exploited the operating system, and ran fast and clean. In short, he was a hacker from the old school.

Darin proved to the managers at Apple that he was very good, certainly good enough to become the lead programmer for a new version of the operating system. There was a new group forming at Apple to put together all the pieces of this software, and Darin was offered the lead position. That group became the Blue Meanies, the integration team for System 7 at Apple. The 13 programmers on the Blue Meanie team were responsible for making all of the pieces fit together. There were over 700 people working on System 7 at one point, and Darin

was responsible for making sure that all of the code they wrote would work together. Darin was totally absorbed in System 7 for more than two years, the weight on his shoulders was immense. For his team, Darin picked seven other MacHack veterans and five seasoned Apple programmers. At the time that System 7 was being written, it was the largest software project ever attempted outside the U.S. government. There were times when the team was fragmented and Darin had to pull them all together again. There were times when they were exhausted, and Darin had to send them home. And there were always deadlines. The Meanies had to deal with late code, poor code, and missing code. The vast majority of the job was deadly boring, integrating other people's well-written code. The hard part was finding the problems before they left. Darin and his team excelled.

Having completed System 7, Darin wanted a change of scenery and a new challenge, and Andy Hertzfeld and Bill Atkinson offered him both. They offered a move to another city and a chance to build a completely new operating language. The company would be funded by Apple and others and would be aimed at the growing communications area of the computer industry. The company was General Magic, and the product was Telescript. General Magic made the first public announcements in the fall of 1993 and the first major demonstration at MacWorld in SanFrancisco in January 1994. AT&T is shipping products with Telescript built in, and many other General Magic partners are building products or services to take advantage of telescript. Telescript is a language that allows you to build agents that then can do intelligent computing for you. Darin is committed to version 2 of Telescript, but there is no telling where he will turn up next.

1988—Growing Pains

The 1987 MacHack conference was a disappointment to many of the sponsors. In fact, only the University of Michigan continued to support the conference. Without the industry support, there was a period of taking stock. MacHack needed both to grow and to find a stable future. One way to do this was to develop the MacHack committee so that the conference would have a larger base of people to

draw from for topics and speakers. Another way was to exploit the growing relationship with APDA in Seattle. In 1988, APDA was a corporation owned by A.P.P.L.E. Co-op in Seattle, and they had a contract with Apple to produce a catalog of programming tools and a magazine. Additionally, they were charged with handling the mail order of programming tools and other developer materials. David Lingwood had again found a number of writers for his magazine at the conference and was willing to provide editor space about MacHack in the magazine. *MacTutor* also published many articles devoted to MacHack.

David Feldt, an Apple instructor, agreed to chair the 1988 committee, and he evangelized it to many people in classes during that year. Darin Adler and Paul Snively of ICOM, Fritz Anderson, Scott Boyd, Doug Clapp, David Lingwood, Eric Shapiro, David Smith, and several others formed the 1988 committee. It was largely through the network of friends and business associates that the 1988 conference came together. The legal problems and the rift with Apple were gone. This lead to a smoother start on the 1988 conference. There were a number of personal problems that slowed down the committee's progress. In many ways, people wondered if MacHack, now an attendance success, was going to be a programming and financial success. Several people went into overdrive to make the conference happen. A couple of them burned out on it. Jordan Mattson single-handedly wrestled critical equipment, speakers, and information from Apple. Again, Jordan marched into the breach and fought for the right thing.

Even without the support of many vendors, MacHack was getting better. To warm up programmers for MacHack there was a five-day Macintosh Programming class at the hotel that proceeded MacHack and a five-day course on database programming that followed. This was a major expansion on the two-day course offered the year before. Many novice programmers took advantage of the warm-up and felt better prepared to participate. There was a track of programming set aside for novice programmers, and many of the senior programmers attended one or more sessions of the novice track to brush up on one or two new topics.

Ted Nelson of Project Xanadu, was a keynote speaker who focused on the future of information and information technology. He was scheduled for 90 minutes, but held the audience for over 3 hours. He talked about a world where information would be available to everyone. That information would be accessed by people and people would then either contribute to the information pool to pay for what they used or they would pay a royalty to the information providers. This was the heart of the Xanadu Project, a project that had been in the works for over a decade and had generated a number of articles and tow books. Ted Nelson was a folk hero to many of the programmers, and his ideas appealed to many more. Several sessions had to be rescheduled. No one minded because Ted was a speaker that everyone thought should be invited back in the future. Ted, an Ann Arbor native, stayed in the area all week and spent many hours with some of the MacHack attendees, several of whom eventually found time to work on the Xanadu Project.

Jordan had worked hard over the past year. Paul Snively, Darin Adler, Scott Boyd, and Greg Marriott (MacHackers all!) were now Apple software engineers. Several other MacHack attendees were interviewing at Apple, and Jordan knew the right engineers to invite to MacHack; he provided that information to the committee. He also provided information on topics that were not yet public knowledge. This information allowed MacHack to be on top of the subject matter. Additionally, Jordan lobbied engineers who were invited to take the time to go. His help brought more technical knowledge to the conference.

The "Bash Apple" session was again held on the first night. It did not carry the intensity of the 1987 session. First, there were now several people on Apple's side of the podium. Second, several of the people who had asked the hardest questions were Apple employees. There was no question that the session was more mellow than the 1987 session, but just the same, the session cleared the air and made it easier for everyone to work together. Again, the two-hour session ran overtime, but like the 1987 session, questions were carefully recorded and answers were provided to all the attendees. The session ended in a different fashion as well—there was a good-natured water fight that spilled into the parking lot.

The machine room expanded from a single room in 1987 to two rooms in 1988. There were a dozen color machines where, in 1987, there was but one color machine. The room was networked, and a laser printer was available to make print outs. The two rooms were each as crowded as the single room was in 1987; people lined up to wait for machines.

There were now three tracks of programming, the topics were lively and timely, and the speakers were becoming luminaries in the industry. The lead programmers from companies like Think (now Symantec) and ACIUS, Texas Instruments and Odesta were on panels. Networking, connectivity, nationalization, and UNIX were very hot topics, and tools were debated in depth by the attendees. A new and novel technology, CD-ROMs, were discussed in detail, and all of the attendees were invited to try their hands at building them. Scott Boyd and Greg Marriott of the MacHax Group decided that with all the programmers in attendance, they would hold a programming contest, so they set up rules and announced the contest at the conference. The people had two days to write a hack, hence, the first sanctioned hack contest occurred. There were 10 entries, which were shown in the machine room at midnight. There were a number of crashes and lots of laughs. The final two hacks were shown Friday afternoon, just before the close of the conference. The hacks ranged from a pop-up menu for multiple disks on the desktop to a programmer's editor. The favorite hack was the slider FKey that moved all the windows for an application to the left or right, up or down together. With Multifinder and small screens on most Macintosh computers, it was easier to find a single application and all of the windows that went with it by moving the applications above it. The authors Bill Johnson and Ron Duritsch later worked on a hide windows and window minimization routine, like windows uses, for the Mac. The hide windows routine became part of System 7.

One of the hacks that runs today on most of the machines was "The Clock", which is a full-screen digital clock, the forerunner of the screen saver. Figure 1.1 shows an example screen from the clock.

FIGURE 1.1 THE DISPLAY OF "THE CLOCK" INIT

The program had grown by over 50% again from 1987. The attendees were thrilled, and the conference was over 200 strong. It was so big that a number of people were worried about the future of MacHack. With strong press in the offing, there was no question that something had to be done about the future. The volunteer committee continued to grow in both size and quality. Word of the Hack contest was spread between programmers, and the decision was made to limit future MacHack conferences to 300 attendees. This was debated by everyone involved. There were some who said the conference would become elitist and others who felt it was too constrained at 300. Some thought that 300 was too large. The final decision was less than unanimous, but it has served the conference well.

Brita Meng from *MacWorld* magazine attended MacHack, enjoyed it, wrote about it, and joined the 1989 committee. This led to an amazing growth in interest from around the world. Brita became a driving force in the future of MacHack.

The MacHax Group Scott Boyd and Greg Marriott, Texas to Cupertino

In 1986, when MacHack first was started, a letter filled with legal jargon was sent to Expotech about TradeMark infringement. This letter took over a year to resolve and was really only resolved when Scott Boyd, the author of the letter, came to MacHack in 1987. Scott was willing to sit down and settle things, and when the settling was done, Scott and MacHack were linked. Scott and his partner Greg Marriott founded what has become the Tradition at MacHack—The Best Hack Contest. Scott and Greg met in college in Austin, Texas. They caught Jordan Mattson's eye early and were invited to interview at Apple; both were hired.

First as members of Apple's Developer Support Group (DTS) and later as members of the élite Blue Meanies, Greg became the manager of testing for the Blue Meanies and Scott became one of the key programmers working on the integration of System 7. Greg went on to take Darin Adler's place as the lead programmer in the Blue Meanies when Darin left for General Magic. After about a year, Greg left and followed Darin to General Magic, where they again worked side by side. Scott stayed with the Blue Meanies until the last Apple reorganization, when he chose to look outside the Apple family of companies for his next position. Neil Ticktin of MacTech convinced Scott to join the magazine as editor, and the first issues with Scott's mark on them hit the stands in March 1994. With a 10-year veteran of the Macintosh at the helm, Scott is planning to take MacTech back into the limelight that MacTutor used to enjoy. Greg on the other hand sat in the front of the hall and watched unveiling of Telescript to a standing room only audience at the San Francisco Marriott in January 1994. Greg is just getting started with Telescript. Meanwhile, both Scott and Greg served on the MacHack committee and oversaw the growth of the Best Hack Contest.

1989—The Macintosh Technical Conference

It was less than a month after the 1988 MacHack when the first inquiry came in from Europe. Apple France wanted to send some of their staff to the conference. In the next four months, over 30 inquiries were fielded from around the world. Apple asked the committee to consider setting up a MacHack Europe. After some investigation on costs, the committee declined.

The 1989 conference was chaired by David Feldman of ICOM Systems. By late fall, there were more than 30 session topics on the table and up for discussion. The committee had grown to over 20 people. This led to the most heavily scheduled MacHack to date. The range of topics suggested meant another leap in the number of MacHack speakers. In many cases, people had to leave active projects for a week to speak at MacHack. This was less of a problem in 1989 than it was in 1988. Apple was on a roll, and the Macintosh was climbing the best seller charts. Other companies in the Macintosh industry were benefiting from the explosive growth in sales. Apple and other companies gradually became convinced that MacHack was the place to find critical programmers.

The year 1989 marked the beginning of the MacHack Paper's contest. Waldemar Horwat suggested to the committee that the conference solicit papers from leading researchers in the field of computing and offer them a chance to present at MacHack. Also, they were solicited from college students at all levels, which worked so well that half a dozen were presented, ranging from *Parallel Processing Paradigms* to *Cryptography*. The paper sessions were jammed. Everyone wanted copies of the papers. Copies of the papers were placed on disk and offered to the attendees. From these papers came new encryption software for the Mac, the first prototype parallel processor Macintosh, and lots of discussion. Waldemar presented the *Parallel Processing* paper and ended up with a number of interested people at MIT. He offered to organize it again, and the committee agreed.

With over 55 sessions, MacHack was very full, perhaps too full! There were three sessions starting at the same time, and the session format had switched from a single presenter to a panel discussion. The panel format was a major negative, which was abandoned as a failed experiment. There was schedule confusion right up until most of the sessions started. The committee (mainly the author) did not do the job that they should have in confirming speakers. Coordination of speakers between the various committee members also added to the confusion. The 1989 MacHack was a learning experience.

Scott Knaster from Apple was the keynote speaker who began the conference on the right foot, getting the audience excited about programming and discussing his time on the Macintosh team. In both cases, Scott had many lessons that he shared with the audience, whom he challenged to always look for new ways to do things. He asked users what they did not like about a computer or software, and then told the programmers to fix what the users did not like, rather than dwell on what they liked.

MacHack spent a lot of energy on tools and programmer utilities, scheduling sessions on MPW, Think C and Pascal, CL/1 (now DAL), MacApp, and C++. In each session, the speakers were either the product managers or the lead programmers for the products. This lead to a very healthy compromise between the speakers and the audience. Beta test sites were established for attendees who made the best suggestions. Over one-third of the attendees at MacHack were also speakers. Informal sessions were held at all hours all over the hotel. Information exchange ran at a very high level, both in and out of the sessions. All of MacHack ran on *Programmer Fantasy Time* (PFT). There was not a day or a session that started or ended as intended. The schedule was out of control. Hotel security repeatedly attempted to make the programmers go to bed, without success.

The Best Hack Contest also was off to a roaring start, with over 50 declared entries. Chris DeRossi, of Developer Technical Support at Apple, won the contest with Color Finder. Chris found a simple way to patch the operating system, so that it would display multicolored

icons. Chris joined the Blue Meanies a few weeks later and added the capability to System 7. There were 34 other hacks in the contest, most of them programmed at the conference. This was up from 10 entries the prior year. The Hack Show was run at midnight, but, as usual, it started late, another victim of PFT. The Hack Show was heavily attended and ran for several hours. The hacks ranged from mundane like ASCII DA, to silly like the Grouch, to bizarre like Fridge Watcher.

Eric Shapiro showed the Grouch (see Figure 1.2) to the audience to wild cheers and laughter. Everyone in the room was familiar with Sesame Street and Oscar the Grouch. Well, Eric brought Oscar to the Desktop, in the form of a singing animated trash can. This became the signature hack, not only for Eric, but for MacHack. There was a downside when it was released to the on-line services. Eric received panic phone calls from parents who had every item on their hard disk thrown in the trash by a preschooler who wanted to watch and listen to Oscar.

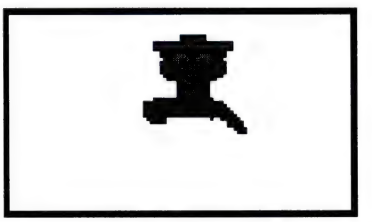


FIGURE 1.2 OSCAR CLIMBING OUT OF THE TRASH CAN

Init was a Control Panel (Figure 1.3) that allowed users to select the INITs (now called extensions) that would be loaded by the computer when it was started again. The concept lead to a number of INIT managers and finally to the Extension manager used in System 7. The major drawback of .Init was the icon (see Figure 1.4), which caused a number of users to try to open the folder that the icon seemed to indicate that it was.

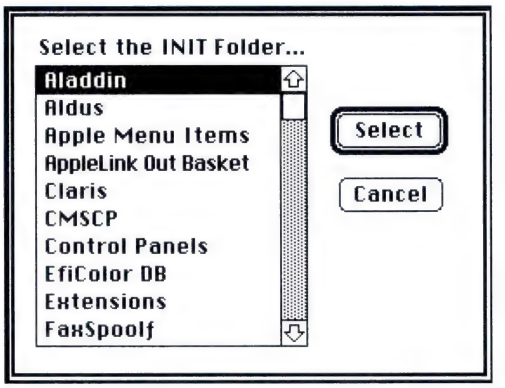


FIGURE 1.3 INIT EXTENTION SELECTION SCREEN

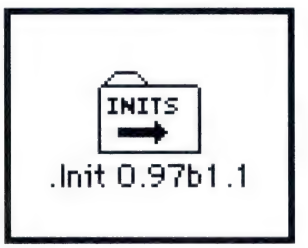


FIGURE 1.4 THE FAKE FOLDER ICON FOR .INIT

Brian Topping created an MPW tool that allowed programmers to send code to another machine. This tool was the forerunner to a number of the network installers that are used to install software over local area networks.

Greg Marriott, a member of the MacHax group, revised his Blast Fkey to allow it to make random-shaped holes. Blast shot holes all the way through documents on the desktop and through the desktop itself.

Buckets was a color painting program written by David Surovell (see Figure 1.5). It led to David joining the QuickdrawGX team at Apple. It was so well written that it continues to run under System 7.1

on a number of machines that did not exist when it was written. The program had a set of tools and colors that were better than any of the shipping commercial programs. It lead to the upgrades of Superpaint and others, in an attempt to avoid having a free program take over that portion of the market.

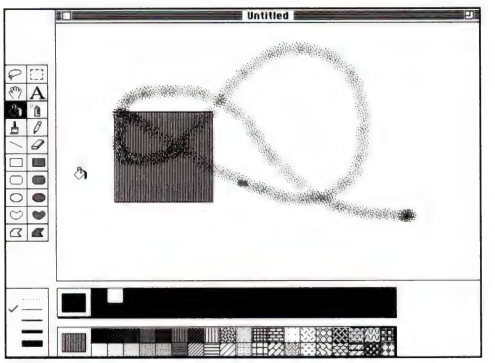


FIGURE 1.5 BUCKETS TOOL PALLET AND PAINTING AREA

David Shayer of Apple created a program that put controls on the screen, (see Figure 1.6) so that programmers could quickly mock up screens for customers. This idea would be repeated in a number of commercial and shareware products that would follow.

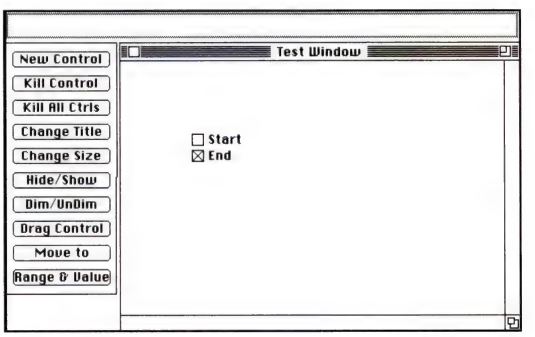


FIGURE 1.6 RADIO BUTTONS PAINTED WITH CONTROL PAINTER

The Apple Tools team did a silly little hack (see Figure 1.7) that put a picture of their refrigerator on the screen and called it "fridge watcher."

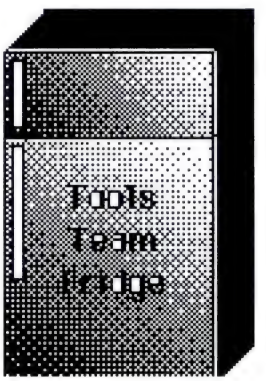


FIGURE 1.7 THE WHOLE INTERFACE OF FRIDGE

David A. Smith (not the same David Smith as *MacTutor*) won an award for the most educational hack with his Robot World, a robot arm simulator (see Figure 1.8).

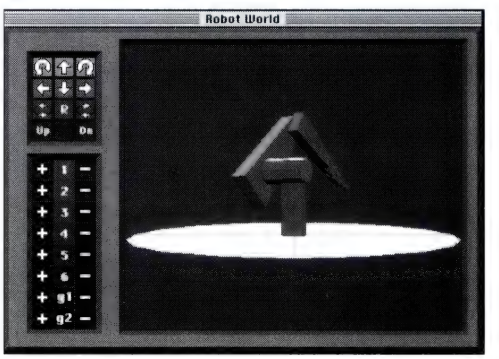


FIGURE 1.8 THE ROBOT ARM, READY TO RUN IN ROBOT WORLD

Dean Yu wrote his first two hacks, a game called Mazer. The opening screen is shown in Figure 1.9; a three-D maze game that still plays on most machines today.

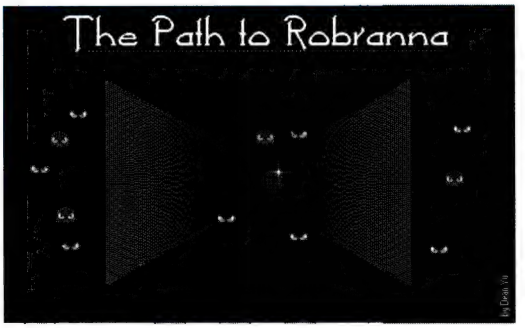


FIGURE 1.9 MAZER WAS THE FIRST PLAYABLE ENTERTAINMENT HACK

The Hack Show was a rousing success, so much so that at 11 AM the next morning a crew packed up many of the hacks and headed for the MacTechnics meeting to show them off. There were ovations, laughter, and more than a few exclamations of excitement. This cemented the relationship between MacTechnics (who provided many of the volunteers) and MacHack.

The Hack Contest would not be possible if Dr. Carl Berger at the University did not authorize the University Computing Center to tear down one of the newest labs at the U of M and transport it to MacHack. This lead to a state-of-the-art machine room with lots of goodies in it to play with.

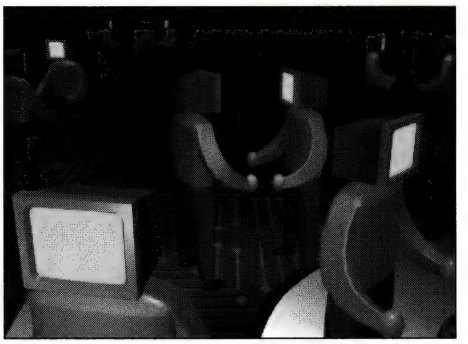
With the need for color projection in all of the sessions and the requirements for better sound, the equipment was more complex. Technical difficulties were the rule rather than the exception in 1989. The job was getting more complex, and the volunteers were also growing in their regular jobs. This was leading to problems, and a solution was urgently needed.

In the weeks following MacHack, Apple hired several more of the attendees. Several of the concepts that were first shown in the hack show, like Color Finder, Window list, and Higher Menus, were included in System 7. For a conference of just under 300 people, MacHack had a profound impact on what Apple and its users were going to see in the next few years.

Dr. Waldemar Horwat—TMON at 14, MIT at 17

Waldemar first attended MacHack in 1987. He was a shy 17 year old at the time. He was also the programmer of TMON, the first true debugger for the Macintosh. Waldemar was working on some Macintosh code when he realized the existing debugging tools would not let him fix his program. He wrote TMON in Assembler, and he wrote the entire program at once, instead of in pieces, unlike most projects. When the program was complete, he assembled it for the first time and then tested it. He made a few changes, changed the version number by ".1," and reassembled it. In just a couple of weeks, Waldemar had written a debugger that was on par with programs that would later require several effort years by Apple and other large companies. In 1989, Waldemar, an MIT student, presented the first papers at MacHack. He chose to chair the paper track from 1989 through 1993. During that time, Waldemar gave eight papers on a wide range of computer science topics. For summer vacation in 1993, Waldemar, a doctoral student, chose to work with the General Magic team to complete the Telescript engine. It reunited Waldemar with Darin Adler for the first time in several years. Waldemar was also offered a position by Apple to work on the 68-K emulator for the Power Macintosh series of computers. Waldemar is recognized as one of the world's top programmers. He has done parallel processing compilers for MIT's Connection Machine, and he has looked at the underlying assumptions about emulation of one computer on another. He has just been awarded his PhD. in Computer Science. His thesis dealt with abstraction.

CHAPTER 2



1990—System 7 Begins to Gel

After the announcement of System 7 at the World Wide Developer Conference (WWDC) in 1989 and the sessions at MacHack that year, the committee was focused on the idea that System 7 would be released before MacHack in 1990. The focus of the committee was on getting sessions about the released software and case studies on products that were either new or updated for System 7.

The conference was loaded with sessions on System 7. A first, Ms. Brita Meng, was the chairperson for MacHack. The committee was streamlined from the previous two years, and that helped the committee to function more quickly. With fewer than 20 people, instead of the over 40 from the last 2 years, the conference was easier to plan and manage. Jordan Mattson arranged for everyone to be included in conference calls once a month, which allowed them to feel like a committee for the first time. These calls were held monthly, with everyone in the loop, and many sessions were planned before the official announcement of products and directions was made.

As the year wore on and there was no news from Apple on the release of System 7, the committee began to worry. Finally, Darin Adler told the group in November that at best there would be a beta of System 7 in the hands of pre-MacHack developers. The group scrambled to

Case studies were out, and coding and porting techniques were in. With the indicated changes in System 7 from 1989 to 1990 and the pull back from QuickDrawGX, there was a need to rethink much of the program that had been developed. Additionally, there was a need for programmers to relearn many of the topics that had been reviewed in 1989. Apple had turned the world upside down once again. Several sessions were held open until after WWDC in May 1990.

Apple's big push at the WWDC in 1990 was System 7.0. As in 1989, Apple promised that System 7 was right around the corner and that everyone needed to focus development efforts on it. They pushed hard for all the programmers to begin to make software that was compatible with it. Apple seemed unconcerned that System 7 was still in alpha testing, and everyone involved with the project was saying that it would not ship until late in the year at the earliest. The software was already more than a year late in shipping, and the programmers were saying privately that it would take several more months to finish a final product. Apple offered a CD-ROM to everyone who attended the conference with the alpha version of System 7 on it. They promised that a beta would be available by midsummer. Most of WWDC dealt with an overview of System 7 for the attendees. Speakers at WWDC spent time on True Type and True Image, Color QuickDraw, being 32-bit clean, the Edition Manager, the Event Manager and AppleEvents, the Help Manager, and WorldScript. Apple devoted over 40 sessions to System 7 at WWDC. The problem was that none of them were hands-on. With the alpha being distributed at the conference, no one had any serious questions to ask about how System 7 was going to work, since there was no chance to test drive the software in advance. WWDC was a big hit, but most of the programmers left shaking their heads in frustration and confusion.

During the six weeks between WWDC and MacHack, the committee discussed the changes to MacHack in weekly conference calls set-up by Jordan. Everyone was working to build the best program in light of Apple's backing away from QuickDrawGX and the other changes that were announced to System 7. Once again, Apple had left a number of issues on the table.

MacHack was the first conference that allowed the programmers a chance to really get hands-on experience with the System 7 alpha and to talk to the designers of the software. With the alpha in hand for over a month, the third-party programmers came armed for bear; the questions were complex and deep. The designers answered numerous questions and made some changes in the final System 7 code based on discussions at MacHack. Pink, the operating system that Taligent is working on, is based on System 7 and the wish list for System 7 and beyond. A large percentage of that wish list was developed at MacHack. 1990 was also the first time that hacks were written for what is now the basis of Apple's Macintosh operating system.

The keynote was delivered by Mitch Kapor, who addressed the growing concern for the freedom of speech; not in newspapers, but rather, in regard to electronic media and bulletin boards. Mitch had just cofounded the Electronic Freedom Foundation (EFF) and was in the process of standing up for everyone's rights. Formed as a nonprofit corporation, the EFF mission is to defend the first amendment with regard to electronic communications. The first cases that the EFF entered were the Steve Jackson case in Texas and the Robert Morris Internet Worm case. Both were ongoing, and Mitch discussed both cases. He made the attendees understand that the key to the electronic future would be the rulings in court cases, not the laws passed by Congress. His concern was the extension of the first amendment to electronic mail and bulletin boards. The keynote was upbeat and offered hope that the 200-year-old constitution of the United States was indeed the relevant document for everyone to still live by. Mitch stayed a few hours and discussed a number of programming topics with the attendees. He proved to everyone that Lotus Development was not a fluke. His insight on large programming projects and setbacks was welcomed by the System 7 team, most of whom were feeling down.

That evening, the Bash Apple Session was hot, the introduction of System 7 left more questions than answers. The new volume of Inside Macintosh (Volume VI) was not available, and the documentation and sample code that was available for most of System 7 was spotty and uneven. In fact, a large amount of it had the label, "This is subject to change" stamped on the pages. After two years of talking about

System 7, major features and parts of the programming interface were still tentative. Porting programs to System 7 was risky; if you made the port, based on the alpha, and nothing changed, you won. If you made the port and something changed, you had to do it all over again. If you did not port at all, you lost. Timing of the port was critical. Companies spent millions of dollars chasing System 7 during 1990. With eight of the 13 Blue Meanies, the System 7 integration team and most of Developer Technical Support at the table in the front of the room, people finally had a chance to get answers. The audience was unhappy with a number of these answers. Also, there were several hot debates about doing the "right thing" and being consistent in the interface. The Bash Apple Session ended with a silly string fight at 2 AM. Most of the tension was now off, because people had a clue for the first time. Several of the third-party programmers and the Apple Engineers went off in teams to the machine room to test things with System 7 and to try out new ideas. Most of them ended up in the Hack Contest, while several ended up as part of System 7. Most of these hacks appear on the CD.

Waldemar Horwat had once again rounded up a group of outstanding papers. His own presentation dealt with C++, an up-and-coming language for programming. His paper, which looked at the power of C++ language extensions and the demonstrations that went with it, caused many programmers to begin the conversion to C++. Shane Looker discussed the idea that applications could be extended by using object-oriented technology. Shane's talk was heavily attended by the Apple employees at MacHack, and it predates Apple's own OpenDoc strategy. Shane did not start the move to extensible programs, but he was one of the first to make the technology accessible to people. Finally, Ralph Krug discussed the idea of pop-up menus using the MacApp development environment. Taken together, these three papers signaled trends that are still evolving in the software arena. Several hacks took advantage of the technologies that Waldemar, Shane, and Ralph discussed, but in many cases, the results did not show up until MacHack the next year.

The leading programmers from the industry presented on a number of key System 7 topics. Tom Evslin spent time discussing Royal, Apple's new font and printing architectures. These became True Type

and True Image, the basis for fonts and printing in both System 7 and in MS Windows 3. Darin Adler and Chris DeRossi from Apple delved into the art of moving code from System 6 to System 7. The new version of system software required that a number of standard programming practices be modified, so that programs would run. A large portion of Apple's own code did not run without modification. System 7 broke most of the rules, so that the Macintosh could go forward. This session was heavily attended, giving many companies a leg up on the System 7 market. Of the 100 or so companies that had the System 7 balloons at MacWorld in August 1991, more than 75% had sent programmers to MacHack. The Communications Toolbox was discussed at WWDC for the first time in 1990, and two of the earliest users of the CommToolbox, Marshall Clow and Leonard Rosenthal, discussed how the CommToolbox would make using modems easier. Bill Britton attended this session and later wrote a program that would become SitComm!, with some help from Leonard. Paul Campbell of SuperMac discussed the slot manager, the magic code that makes the Macintosh plug and play for every manufacturer. This means that the Macintosh does not need to worry about interrupts and other user settings like the Intel-based PC's. Interapplication Communications (IAC), the forerunner of AppleEvents, AppleScript, and the Apple Open Collaborative Environment (AOCE), was presented by a panel of experts. Color 32-bit QuickDraw was a hot topic for many, and several programmers had their first chance at MacHack to try it on a color machine.

The machine room, stocked by the University of Michigan was filled to the brim with color Macintosh computers and other hardware goodies. This is the first year that the dominant machine was based on the Macintosh II architecture. It was also the first time that the network was Ethernet instead of LocalTalk. The change in network design resulted in a number of Hacks that took advantage of the faster network. NetBunny was just one of them. The machine room also had an overshadowing problem for the first time: *Virus* writers had come to the Macintosh. Don Brown from CE Software and John Norstad from Northwestern University, both MacHack veterans, had developed programs to combat virus problems, but the door had been opened and computing was no longer a safe occupation. The first thing most

of the programmers did when sitting down at a machine was to install a virus checker and to double check the machine for viruses. Normally, the machine room was two deep in people waiting to get at a Macintosh. In several cases, it led to a discussion that led to a hack, which has led to a product or a job.

Before the Hack Contest on Thursday night, at the banquet, Eric Shapiro had his chance to display the first MacHack food Hack: fortune cookies with sayings in them that were either from earlier MacHacks or from other computer sources. The attendees enjoyed the fortunes, some of which are repeated as follows:

- Unix reacts to the user as though it is being annoyed.
- A lean compromise is better than a fat lawsuit.
- "What, no basic in ROM?" —*Scott Knaster.*
- May your INITs never conflict.
- "Daddy, if you stay at IBM, can we still keep the Mac?" —*Apple V.P.'s daughter.*
- It's all Jordan's fault.
- The only difference between a necktie and a noose is that the noose is quicker.
- MacApp's method dispatch delay is an inheritance tax.
- Thou shalt not use nil handles or nil pointers.
- Thou shalt not write code that modifies itself (except when storing A4/A5).
- Thou shalt not create or use fake handles (except if you're MultiFinder).
- Thou shalt not assume the screen is a fixed size.
- "Reserved" means "do not touch."
- When really in doubt, blame Jordan and the Hardware.

- 32-bit cleanliness is next to godliness.
- Never trust anyone who says "Real Soon Now."
- "Don't call us, we'll call you" —*MacApp Development Team.*
- Never call DisposHandle on a resource.
- "We are not authorized to give you that information." —*MacDTS.*
- Tomorrow's version of yesterday's hardware is 'Bad Karma' —*John L. Gassee.*

The conference continued at a fever pitch all night, through the hack contest and into the next day. Friday was a fast-paced day filled with sessions. The final session of the day was the hack awards.

Dean Yu, a student at the University of Michigan and a prolific programmer, spent most of MacHack designing and building a set of INITs or extensions to the Macintosh interface. While in concept the programs were very good and innovative, none of them had any real testing; hence, most users tended to crash when they used these programs. Dean is famous for NetBunny, a program that, once installed on all the network machines, will send a pink bunny with a large base drum running from screen to screen on the network. NetBunny came to symbolize the Hacker ethic for many programmers. Because of the problems with Dean's INITs, there was a program written that would find them and shut them off. This program is INIT31+, and it is very proficient in removing the pesky INITs that Dean wrote as a student. Dean has since gone on to write some of the most solid code in the Macintosh operating system for Apple. Dean, like many programmers, worried more about innovation than testing in the early part of his programming career. Since then, Dean has learned about testing and maintenance. For all of Dean's efforts, he won a number of best and worst hack awards at MacHack. The prizes, as usual, included old textbooks, out-of-date software, and a rat trap. Many other silly prizes were also awarded. There was no cash award, no trophy, just the acknowledgment by peers of a job well done (or poorly done).

John Norstad

John Norstad is a computer programmer, but in reality he is a respected teacher and hunter. John hunts a very special game—computer viruses. When the SCORES virus was released, Northwestern was one of the first sites it hit. John was involved in networking and supporting the on-campus computing environment. He and several other experienced programmers determined that the problems they were seeing on various computers was behavior that was similar to the behavior that had been reported in the DOS community: that of a computer affected by a self-replicating computer program or virus. SCORES was the first Macintosh virus that spread any real distance. It was quickly spread through all the on-line services and the Internet to almost every Macintosh site in the world. Because there had been no virus problems before, SCORES, similar to a new human virus, spread like wildfire.

Don Brown, of CE Software developed a program called *vaccine*, which stopped the SCORES virus from reproducing, but did not remove it. It did, however, tell you that SCORES was active on the computer. John took a harder road, not only of finding the virus, but of removing it and repairing the file damage that it had caused. For several months, this program was the focus of John's efforts. It was released free of charge to the computing world. It was excellent and could be distributed by anyone. John included a text file with the program that explained what a computer virus was and how to tell if you had one. Disinfectant, John's program, became the standard that everyone else measured themselves by. Jeffrey Schulman, the author of Virus Detective, is John's good friend, and they cooperate on the efforts to hunt down and understand new viruses as they are turned loose on the Macintosh world.

John has made Disinfectant his life's work, or so it would seem. Northwestern has supported his work and has made it possible for John and the band of programmers that assist him to continue to update Disinfectant every time a new virus is discovered. Normally, the time span from identification of a new virus to the release of a new version of Disinfectant is less than 3 days, which is incredibly fast pro-

gramming. Only once has the version of Disinfectant that was released had to be replaced with a new version because of a bug. If the rest of the computer industry were this good, there would be far fewer crashes.

John is a long-time attendee and speaker at MacHack whose talks have centered on application design and human interface design. John has created several example applications and a couple of libraries that offer programmers a chance to get a leg up on Macintosh programming. They also offer very well tested, solid code that most programmers can just plug into their application.

John has turned his eye to the *Internet*, a forerunner to the new information superhighway. He has created a group of tools for the Macintosh that make reading and answering electronic mail and news on the Net more fun and easier to understand. The core of them is newswatcher.

John's greatest contribution to the programming community is the training that he has provided. As a lecturer at Northwestern and also as a speaker at a number of computer conferences, John instills the idea that code needs to be well written and tested. Solid design with great testing leads to code that will continue to run for many years to come.

Notably, he has won a number of awards at MacHack for both Disinfectant and for his other programs. Also, he has been recognized for his efforts by Apple and the computer press.

John has turned down a number of very good job offers, because he enjoys teaching others how to do it right. He teaches because he can and not because he has to. Now if he would only learn that roller coasters are fun and usually safe.

The 1990 Hacks

The year 1990 was the first time at MacHack that the programmers had access to a version of System 7. This is the version of the operating system that most Macintosh users run. Because most of the earlier hacks are based on a version of the operating system that has fallen into disuse, the hacks from 1987 through 1989 have not been included

on the CD-ROM in the back of the book. The 1990 hacks were all written to a prerelease version of System 7, in fact many of the features in System 7 changed between MacHack in June 1990 and May 1991 when System 7 was released to the world. These hacks are old and predate almost all of the current generation of Macintosh computers. If you are using a 68040-based machine (Centris or Quadra), turn off the Cache in the Cache Switch Control Panel. Also consider turning off 32-bit addressing in the Memory Control Panel. Neither of these features existed when these Hacks were written, so use these hacks with caution. In Chapters 3 and 4 are newer versions of many of the hacks that are listed as follows. Several of them were written specifically for System 6.x and should only be run with System 6.x. Again, if you are running System 7 and you encounter a problem with the hacks, hold down the **Shift** key and restart. Continue to hold down the **Shift** key until you see the **Welcome to Macintosh** screen with the note on it "Extensions Off."

!Multifinder by Dean Yu

6 7 ~~040~~

FIGURE 2.1 THE SYSTEM 7 PULL-DOWN MENU

System 7 changed the way that the icon at the right end of the menu bar worked. Previously under multifinder the icon could be clicked on, with the mouse, to move from application to application. This was changed to a menu system (see Figure 2.1) under System 7. Many users of System 6 found that they preferred the old way of doing things to the method in System 7. So, Dean wrote !Multifinder (not Multifinder) for those people. In addition to the click to change applications, **Option-Click** will hide the windows that do not belong to the new application, again keeping the functionality of the old version of system software. To use !Multifinder, just drag it onto the system folder and then restart. To move between applications, just click on the icon in the menu bar with your mouse. To hide all the other application windows, hold down the **Option** key and then click the mouse on the icon. While !Multifinder did not break any new programming ground it did offer, the first hint that a number of the new features that Apple was offering in System 7 were not universally liked.

Talking Barney

7 ~~040~~

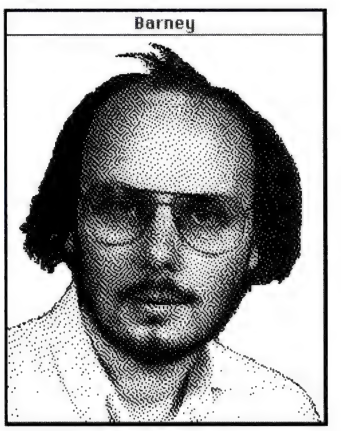


FIGURE 2.2 TALKING BARNEY IN ALL HIS GLORY—WHAT HE SAYS IS ALMOST AS SILLY AS HOW HE LOOKS

With the advent of the new sound manager in System 7, there was now a way to play digitized sounds without having to resort to third-party code. Talking Barney was a group effort to test some of the functions of the sound manager. It is a very silly, but very useful, illustration. Be forewarned that leaving Barney open on your office computer will cause people to talk about you. Talking Barney is definitely not politically correct.

Breakout Screen Saver by Keith Stattenfield

6 7 040

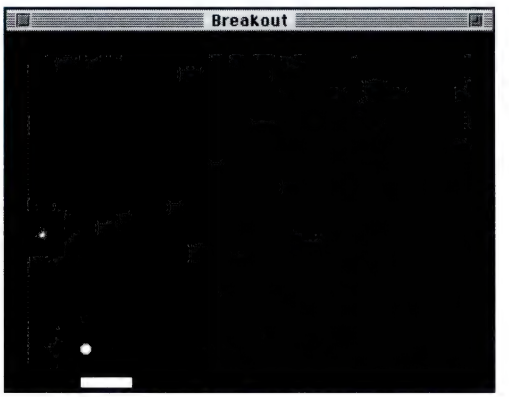


FIGURE 2.3 THE COMPUTER HAS BEATEN ITSELF AT BREAKOUT

In January, at MacWorld, After Dark was released to the Macintosh community. Keith was one of the first programmers to start creating modules for it. Breakout or Break the Bricks was a very old video game, so Keith decided to update it for the Macintosh and to create a Breakout module for AfterDark. Keith included a menu to control the speed at which the game played itself. On some of the newer machines, even the slowest setting is too fast. To use Breakout, you need either AfterDark or Twilight Zone. Breakout plays itself as shown in Figure 2.3. People cannot sit down and play this version of breakout; the computer has all of the fun instead.

Busy Box by Mark Dalrymple, Larry Lipsmeyer, and Dan Gilliam

6 7 ~~040~~ §

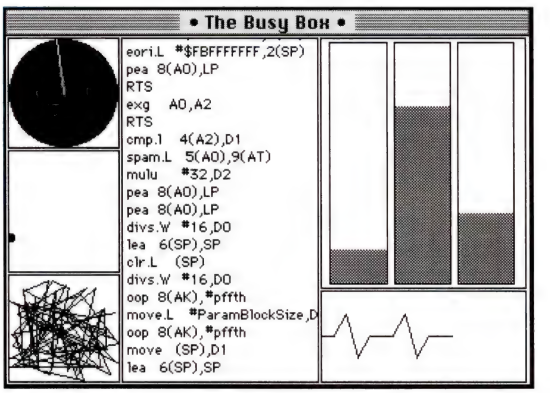


FIGURE 2.4 THE FAMILIAR BUSY BOX FROM THE CHILD'S CRIB TO DIGITAL

Mark, Larry, and Dan were university students who attended MacHack for the first time. They decided to enter the hack contest late in the day on Thursday, after one of them had an idea. After sitting through the sessions on QuickDraw, they wanted to test some concepts. This lead to the building of an application that reported the QuickDraw calls that were being made. Finally, they decided to have fun with it and started to create Busy Box. The idea is child's play, literally. The interesting thing about this hack is that the QuickDraw calls that the program is using to update the screen are listed in the center box. While it looks very different from the commercial busy boxes for a crib, Busy Box is a great tool for learning about the QuickDraw process. A number of people have used the Busy Box code to build QuickDraw debuggers into their programs. This hack was written in Think C version 4 and the source code is included on the CD-ROM.

Eric's ColorWheel Hack by Eric Shapiro

6 7 ~~040~~

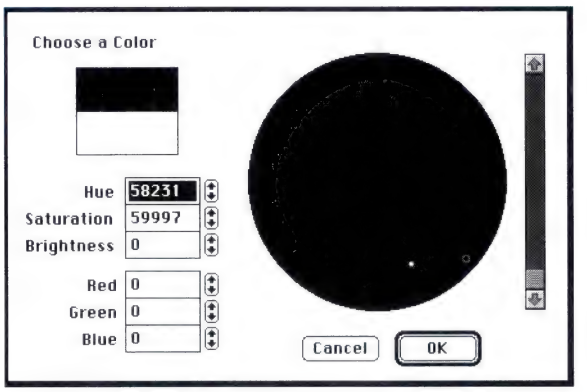


FIGURE 2.5 THE COLORWHEEL HAS A MIND OF ITS OWN

After Eric wrote Oscar in 1989, everyone wondered what he would do to top it in 1990. Eric decided one day that animating the color wheel would be fun. The extension just drops into the system folder and with the next restart, the color wheel has a mind of its own. This was too tempting for most people to use as a practical joke, so Eric created an application that did the same thing instead and left the system alone. Eric had learned a lot about how dangerous silly little extensions could be from a year of having people call about Oscar.

DataStack Filer by Keith Nemitz

6 7 §

The *DataStack Filer* is a library that provides the features of HyperCard cards, in the context of useful programming languages. It was written to translate a SuperCard or HyperCard stack into a real application. This allowed Keith to create Application prototypes

quickly in SuperCard and then move them into MPW C or Modula-2 very quickly. There are two components to the DataStack Filer—*DataStacks* and *StackFiles*.

The DataStack module provides utilities that create a memory-resident stack of cards for records or structures. These utilities have a name and a unique ID associated with them; all of the necessary routines to create, modify, and dispose the stack are defined; and OS errors and self-defined errors are reported. This then becomes the framework of an application.

The StackFile module provides utilities to associate a DataStack to a file. The defined routines create a file, restore an existing file, save the file, and close the file. Saving and restoring the file automatically compacts the stack and improves performance. All OS errors are returned, and files are automatically backed-up each time the file is saved. The file is actually closed until it is saved.

Collectively, these libraries offer a quick way to move a prototype application written in HyperCard and to turn it into a completed application without having to replicate the functionality of HyperCard. Modula-2 source code and MPW objects are provided, but there are no examples of the library included.

Desktop Secretary by Tom Lippincott

6

Tom was a programmer at Software Ventures and an active member of the Berkeley Macintosh User's Group (BMUG) when he wrote Desktop Secretary. There was a problem with the CD-ROM that BMUG was creating—too many files on the disk. The Macintosh file system in System 6 had a lower limit for the number of accessible files than the number of files that could be placed on a disk. This problem was addressed in System 7. There was a fix in AppleShare: Desktop Manager, which Apple wanted a licence fee for. BMUG did not want to pay the fee for every copy of the CD-ROM. The problem was fixed in the new version of System 7, so Desktop Manager was not required;

however, the folks with System 6 were stuck. Tom heard about the problem and wrote Desktop Secretary for BMUG. Desktop Secretary even includes the logo for BMUG into its icon. This hack is still very useful today for people who run System 6 and need to access large CD-ROMs, or have added a few too many files by accident to a hard disk.

Drag Window by Ricardo Batista

67 §

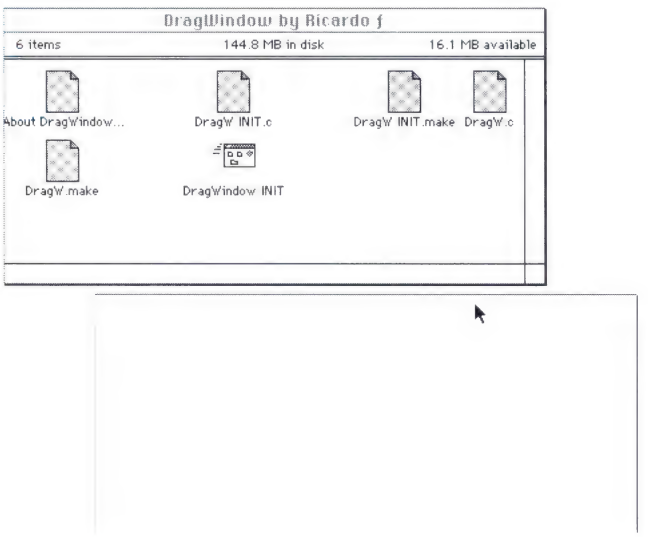


FIGURE 2.6 DRAGGING WINDOWS WITHOUT DRAG WINDOW INSTALLED

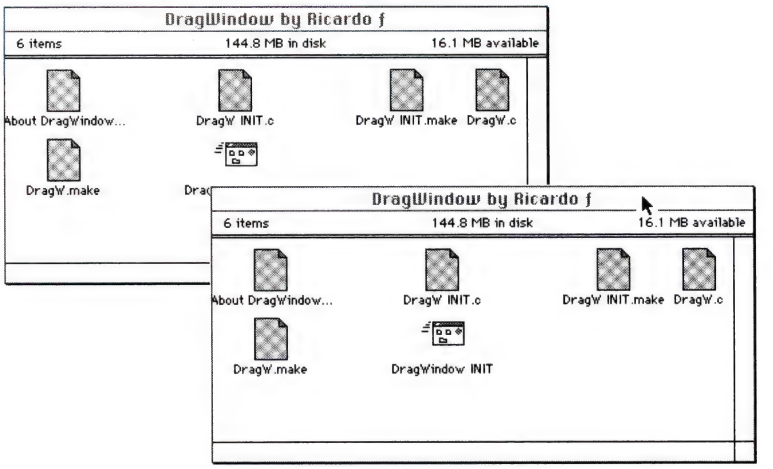


FIGURE 2.7 DRAGGING WINDOWS WITH DRAG WINDOW INSTALLED

When a window is dragged from one location to another on the Macintosh, only the outline of the window shows on the screen. With Drag Window, the window itself shows as it is being dragged. This allows more control over the final position of the window. There is more information to update, so the window appears to be rippling or jumping around on the screen slightly, but when precise positioning of a window is important, Drag Window can save much time and effort. Drag Window was written in MPW C. This hack will finally find its way into the operating system as part of the Appearance Manager in System 8.

DynaRamps by Jay Zipnick

6 7 §040 HW—Requires a color monitor

FIGURE 2.8 A GRAY SCALE VERSION OF A FULL COLOR DYNARAMPS SCREEN

Jay and his brother founded ICOM Systems, makers of TMON, the Adventures of Sherlock Holmes, and Deja Vu. Jay was looking for an interesting hack that would allow him to play with color. With the advent of After Dark and the sales explosion that it caused, several programmers at MacHack were very interested in screen savers. DynaRamps is one such attempt. Jay decided to play with color in much the same way that a number of psychedelic effects were created during the 1970s rock concerts. According to Jay, DynaRamps will either cause you to mellow or become sea sick; it includes numerous patterns and effects that the user can set. Each pattern or effect can be animated by the program, which was written in 68000 Assembler.

Eric's Apple Menu Hack by Eric Shapiro

7 ~~040~~

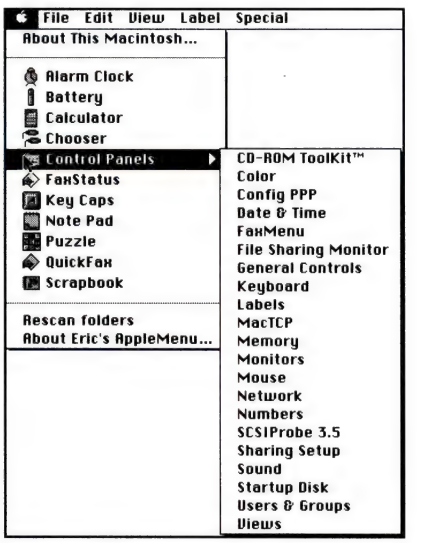


FIGURE 2.9 THE APPLE MENU WITH ERIC'S MENU HACK INSTALLED

Eric Shapiro and Chris DeRossi both had the same idea in 1990. If you could put anything you'd like into the Apple menu, why not make it easy to choose something out of a folder in the Apple menu. Chris created Ham, which was sold by MicroSeeds for several years. Eric created this Apple Menu Hack, which was available on the on-line services for about a year. Apple has decided to implement the feature as an option in System 7.5. Notice that the Control Panel folder appears as a submenu in Figure 2.9, rather than an open window on the screen. This is useful, if you want to get to an item very quickly or are in a low memory situation.

Finder Keys by Jörg Brown

7-040

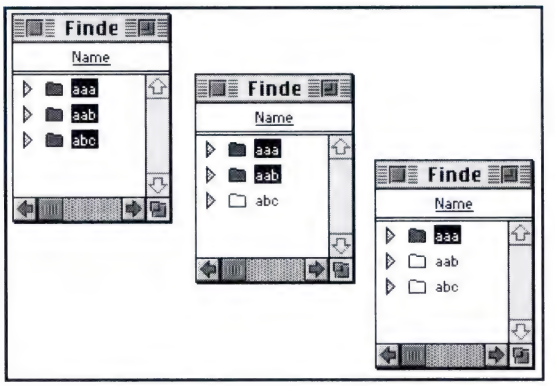


FIGURE 2.10 SELECTING FILES WITH FINDER KEYS

Jörg really liked the idea of being able to type a letter when a window was open in the Finder with System 7 and of having it go to the first icon that corresponded with the letter typed. But that did not go far enough, so Jörg decided to write an extension to System 7 that allowed the user to keep typing. The first letter typed highlighted all the icons whose name started with that letter. As the typing continued, the Finder would narrow the search down to the only item whose name started with the letters typed. In Figure 2.10, notice the results of typing:

```
a a a
```

As the letters are typed, the number of items highlighted decreases until only one item is left. In a folder with 400 items in it, it might take typing the name's first four or five letters to get a single file selected. This was very useful, but it had a problem—nothing could be renamed when the extension was in use. There is an escape that allows about 6 seconds to rename a file: use the **Escape** key to activate renaming a file. Again, you only have 6 seconds in this version. Jörg has refined this piece of code for use in the Now Utilities package from Now Software. The version of this, included on the CD, will only run under System 7.0.0, not under System 7.0.1 or System 7.1.

FinderPict by Chris Derossi, Greg Marriott, Dave Feldman, and Sheila Brady

7 ~~040~~

Chris and the rest of the MacHack System 7 team created FinderPict, so that people could add pictures to their open windows. All it took was having a file in the window that needed a background picture. That file was named FinderWindowPict, and it needed to be a PICT file, created in some sort of drawing program. FinderPict is a lot of fun, but it is also useful when you have children who are too young to read. The picture in the window can tell them what program they are about to use by way of a picture of a rabbit (for Reader Rabbit) or a pirate or whatever. Unfortunately, FinderPict has not been updated in a long time and the program only runs under System 7.0.0.

Grabber by Michael Kahl

7 §

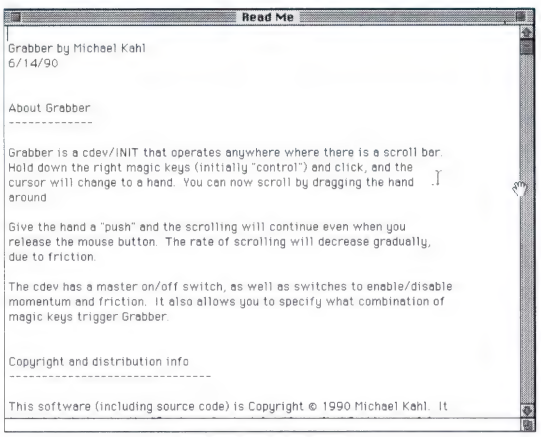


FIGURE 2.11 THE GRABBER IN ACTION

Michael Kahl is the original developer of Think C. When Andy Hertzfeld developed Servant, Michael found that the idea of a hand (see Figure 2.11) to grab things with was very appealing. When Apple did not put the hand in System 7, Michael build a system extension to implement it. He added a couple of nice features that were not in Servant. First, you can enable and disable the Grabber in the Control Panel for the Grabber from the Apple menu. Second, the hand only appears when you are over a scroll bar and when you have hit the magic key. You set your own magic key combination in the Control Panel. Third, Michael allows you to push a scrolling area, and the area will keep going. If you had a 30-page document and wanted to find a heading, you could push the window and the text would scroll by with no further pushing. The text eventually slows down on its own, because of friction, a feature that Michael added. Grabber is written in Think C.

INIT31+ by Jim Wolff and Jim Merkle

6

The many INITs that Dean wrote at MacHack in 1990, spread across several machines in the machine room. Most of them were not debugged. Many of them conflicted with themselves, which caused a number of machines to crash at critical times. The watch word in the machine room was "Kill Dean's INITs," in fact, Dave Koziol and Mary Lynn Sanford had buttons made to that effect. Jim Wolf and Jim Merkle took another approach. Late Thursday, they realized that Dean's many INITs were causing problems, so they went on the attack. It took all night to finish this program, which is installed directly into System 6 software with the installer and modifies your operating system permanently. It compares the name of the INIT trying to load and refuses to load any that Dean wrote at MacHack 1990. This software will not work with System 7, and it can only be removed by reinstalling the system software. It is, however, effective in stopping all the INITs that Dean wrote at MacHack in 1990.

Midi-Driver by Chris Marshall

6 7 §HW—Requires a MIDI device with a Z3580 interface

With MIDI instruments for musicians becoming more common, many people wanted to use the instruments on the Macintosh. A number of the instruments employ the Zilog Z3580 chip. Chris Marshall wrote and debugged a driver for the Z3580 on the Macintosh that made it very simple to add a musical keyboard to the Mac. The driver is the software that allows a computer to talk to something else. There are drivers for printers with names like "Laserwriter" that are found in the

extensions folder under System 7. This Driver is written in C and is very well commented. A number of commercial products have started from this code; however, it does not work with the new Apple AV machines.

Mount Image by Steve Christenson

67

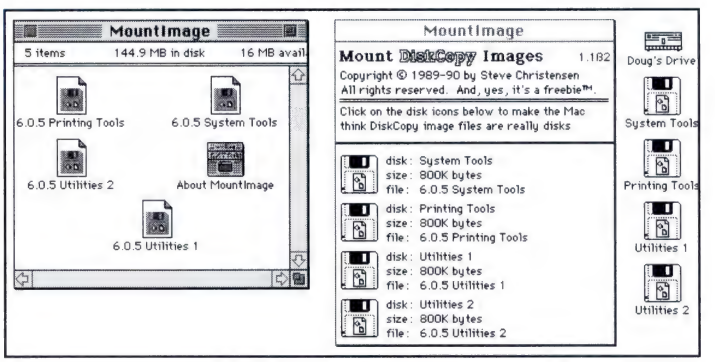


FIGURE 2.12 MOUNT IMAGE FROM DISK FILE TO MOUNTED VOLUME

Steve Christenson is the Apple engineer responsible for Apple's Disk Image software. *Disk Images* are files that are created using the Disk Image software, which are exact copies of floppies. They are useful in storing the original floppies from commercial software. He had a problem installing software, when there were no blank floppies handy to make copies of the disk image files from the hard disk. He also felt that running the disk images from the hard disk would make system software and other installation processes faster. Steve solved the problem by creating *Mount Image*. Looking at Figure 2.12 from left to right, there are the System 6.0.5 disks as images in a folder, the Control Panel for Mount Image, and then the disk images as mounted volumes in the Finder. This system of using the image files directly allows a user to get to software quickly and work with it. The disk image files may be opened like regular floppies, and programs can be run directly from

them. Using Apple's Disk Image program to make Image files of master floppies from software and then backing up the images to tape becomes very attractive. Steve has continued to work on and improve Mount Image, since MacHack, and the current version can be found on America On-Line (AOL) and other electronic networks.

NetBunny by Dean Yu

6.040

FIGURE 2.13 THE NETBUNNY

Dean's best hack at MacHack was NetBunny. Taking his cue from a set of television commercials, Dean created NetBunny shown in Figure 2.13. The bunny when installed on a network of macintoshes, will run from computer to computer (at least one developer used it to call meetings and signal that lunch was ready). To install the NetBunny on a computer drag, the bunnyINIT into the system folder and restart the Macintosh. If you want the rabbit to cross your machine's screen, the bunnyINIT must be in the system folder. NetBunny only works with System 6.0.5 through 6.0.8. Trying to run NetBunny with any other version of system software will result in a crash. NetBunny will work on both black and white and color monitors, and it will run only on the main monitor, if two or more monitors are installed.

NetBunny is unstable, so do not leave it installed. Once it is installed, double-click on the StartWabbit application to start a bunny around the network. If you want multiple rabbits, then continue to double-click about every minute.

The rabbit moves around the network in the numerical order of the LocalTalk socket numbers, so it is a useful tool if your network is flaky. All you have to do is to follow the loud base drum as it moves from screen to screen. The bunny will run until it has crossed all the Macintoshes with the bunnyINIT installed on them once. One of the problems with NetBunny is that it does not always stop after the first time around. The easy way to find the rabbit is to listen for its booming on the base drum. NetBunny will run on a single Mac, and children love to run the rabbit.

Norstad Hack by John Norstad

6 7

The Norstad Hack is the original version of the Disinfectant INIT, which is used to notify users that their computer has a virus. This is not the version to use, it is being included here for completeness. The correct version is in Part B on the CD-ROM.

Note that the latest version of Disinfectant is available on-line from AOL or your favorite electronic service. Please always use the latest version, since new viruses are not caught by old versions of Disinfectant.

Picker Placer by Eric Shapiro

6 ~~040~~ [with this version]

With System 6, the Color Picker always opens on the monitor with the menu bar. If that monitor is black and white, it is very hard to pick the color that you want to use. Eric created a small INIT that kept

track of where the Color Picker should open. This allowed him to pick colors on the color monitor, rather than on his black and white monitor. While this is a simple hack to understand both in concept and execution, it shows one of the problems with Macintosh programming. Only on the Macintosh can you have multiple monitors, out of the box. Programs need to be written to deal with the user's preferences, which include where the window should open, how big it is, and what should appear in it. Most programmers have gone to a preferences file in the system folder to keep track of this information. With System 7, Apple finally began to do this sort of preference tracking.

ShowColor v1.0 by Sean Parent

6 7 040

FIGURE 2.14 SHOWCOLOR PIECES

Many applications written in 1988 through 1990 did not take advantage of the PICT2 format, which allowed PICT images to be full color. Sean Parent built an INIT to make all applications that could display PICT1 images able to display PICT2 images. It did not make it possible to edit the images in a program that was not designed to edit PICT2 images, but it did allow them to be displayed in a document or as a file in the application. Sean also rewrote the scrapbook file to take advantage of the PICT2 format. This rewrite of the Scrapbook was the basis for the Scrapbook in System 7. Finally, the scrapbook was able to display the color images that were placed in it, in color. Place the ShowColor v1.0 and Scrapbook files in your system folder and restart to use ShowColor.

Be forewarned that you will loose all the information in your scrapbook, when you do this. To avoid loosing the information, copy out the items you want to save into files and save them. Then copy them back to the new Scrapbook and dispose of the files that you created to save them.

ShowColor is only required with older applications that were not written to handle Color PICT2 images. The Scrapbook replacement should not be done with System 7 or higher, as the scrapbook that ships with System 7 is capable of displaying color pictures in color and more.

ShrinkToFit by Meredith Lesly

6 §

When a programmer is actively developing a program, there are a number of windows that are open at one time. Getting the windows to the right size, so that the useful information in the window is using the least amount of screen space, can be time consuming. Meredith decided to create an INIT that made getting windows to the correct size with Think C easy to do. ShrinkToFit allows a programmer to highlight a portion of code, and the window will shrink to the minimum size that will display the highlighted section of code, when the **Control** key is held down at the same time the mouse is clicked in the **Grow** box of the window. To use ShrinkToFit, place ShrinkToFit in the system folder on a Macintosh running System 6 and then restart. Using Think C versions 4.x to 5.x, the windows will now resize based on the code that is highlighted. This allows the maximum number of windows to be open on the screen at one time.

Speed Shifter by Pace Bonner

6 7 ~~040~~

Programmers use the fastest machine available to write code, because moving the program over to a slower machine for testing can

be a time-consuming process. Speed Shifter is an FKEY that allows a programmer to slow down the machine that they are working on so that it acts like a slower machine. Speed Shifter is set to FKEY 7 (F7) and offers choices from an SE through Macintosh IIci. This program has had extensive use with Macintoshes running System 6, but it does not always work with System 7. However, it does not seem to crash the machine when it does not work. Remember to reinvoke it to speed up your machine again or to restart the Mac after using it.

SuperGraphics by Eric Iverson

6 7 ~~040~~

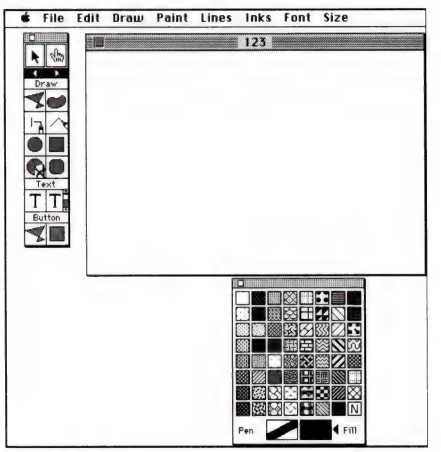


FIGURE 2.15 SUPERGRAPHICS PROGRAM INTERFACE

Eric Iverson is a great programmer, whose first love is SuperCard, the HyperCard look alike that Silicon Beach developed. Eric used the tools in SuperCard to build a full color painting program. This program runs under both System 6 and System 7.

One note if the monitor with the menu bar on it is black and white, SuperGraphics will only run as a black and white program.

In Figure 2.15, the tools and the patterns that SuperGraphics supports are displayed. While this program is written in a nonstandard programming language, it has all the functionality of MacCheese or SuperPaint. Eric spent a week writing SuperGraphics, while MacCheese took several months to complete. The power of SuperCard still has not been fully tapped by most programmers.

Surovell Stuff by David A. Surovell

6 7 ~~040~~

Paint—the latest copy of Buckets, an 8-bit paint program originally by David A. Wilson (of MacApp and Developer University fame) and hot-rodded by David Surovell runs well with System 7 as well as System 6.

GeeWhiz—6 7 ~~040~~ § An example of how to use GWorlds, along with enough source code to read and write PICT files painlessly. GWorlds makes graphics appear on-screen as whole pictures, rather than being drawn a bit at a time.

VBL—6 7 § routines for setting up and installing a vertical retrace task. Every 1/60th of a second the Macintosh takes a break from drawing on the screen to allow the screen to reset to the top of the monitor; this is called the *vertical retrace.* During this time, the Mac is able to continue to function and do background tasks. The code supplied is a library that supports background tasks in the vertical retrace time period. Tasks assigned during vertical retrace should be of very limited duration.

Polly—6 7 ~~040~~ § routines for generating regular polygons, either with or without math coprocessor assistance.These routines are used in a number of Macintosh games to allow the graphics on the screen to take on a 3-D look and to redraw very quickly.

GDevice Hacks—6 7 G'Night: a shutdown application. Hold down the mouse button to see the visual effect without shutting down.

GDCleaner—6 7 ~~040~~ an application that forces a redraw of the desktop; useful when your last application left the desktop only partially

redrawn. Fixes all the problems with killing an application by forcing it to quit.

The Grouch by Eric Shapiro

6 7 ~~040~~

Eric received so many letters and phone calls about Oscar from 1989 that he was determined to keep Oscar alive. There was another company that had a software product with the name Oscar, so Eric agreed to change the name of the INIT to The Grouch. Version 2.0 was designed to run with System 7. Also, The Grouch now sang two segments from the song. The Grouch is not included on the CD-ROM by way of an agreement between Children's Television Workshop and Electronic Arts. The Grouch has since been retired, but some user groups still have it in their libraries.

Vector

6 7 ~~040~~

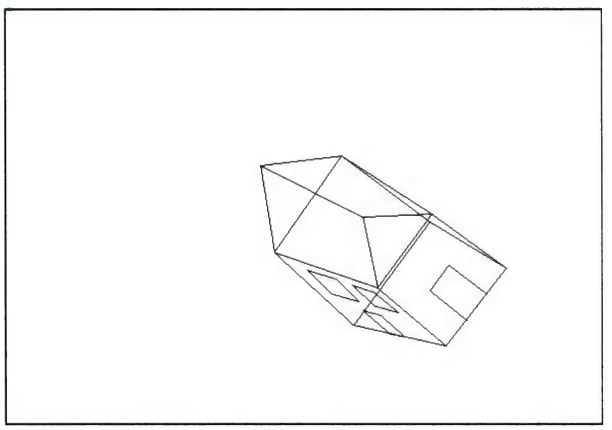


FIGURE 2.16 THE HOUSE THE VECTOR ROTATES

Vector is a hack based on a 3-D library, written to show how the library could handle the rotation of a wire frame house. In Figure 2.16, the house is in rotation. The goal of the library was to make 3-D drafting applications simple to build for the Macintosh. The designers of the library have since introduced a number of deck and room design packages that can be found in Kiosks at Lumber Yards and in Hardware stores.

Volume Dock by Dean Yu

6 7 040

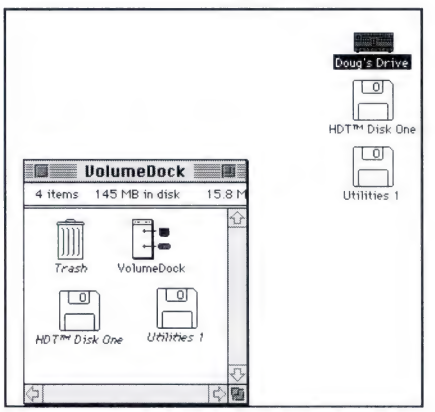


FIGURE 2.17 THE VOLUME DOCK WINDOW

Volume Dock (another hack that Dean produced during MacHack), creates a folder on the top level of the startup disk with the name Volume Dock. Once the folder is opened, you have a window with the trash can and all of the mounted disks in it. This can be arranged so that when the finder is brought to the front, the window with all the disks in it is easy to reach; no more hiding windows or resizing application windows to get to another application or document that needs to be opened. With System 7, volume dock uses aliases to get to the volumes. Notice in Figure 2.17 that other items that are frequently accessed can also be added to the window. Volume dock is another option for computers with very crowded Apple Menus, or that lack the screen real estate to allow lots of items on the desktop.

CHAPTER 3

1991—NetBunny 2½

With MacHack over, most of the committee returned to creating products for the Macintosh. Everyone left with the feeling that the beta version of System 7 would be in their hands in just a few weeks, but summer and fall slipped away and still no beta disk. "SuperBeta" hit the first week in March. The general feeling within the MacHack committee was that System 7 would not ship in 1991. The committee was wrong! The Blue Meanies went into high gear. They missed the conference calls from February until WWDC in May, and none of them answered electronic mail.

Planning the program for MacHack was not much fun; no one was responding to messages. The program was in shambles. Brita Meng, the MacHack chairperson, convinced Jordan Mattson to find Apple speakers and to fill in the program; it worked. Jordan was his usual dynamic self. Popping into meetings throughout Apple, Jordan carried the flag. He was getting names and slotting them against program items. He was more dynamic within Apple on MacHack's behalf than he had been in previous years on Apple's behalf at MacHack. Jordan also coordinated the conference calls. He was busy with new versions of all the products that he managed, but still found time to assist with MacHack. The whole program, as well as System 7 were precarious until the week of WWDC.

There were rumors that Apple had a release version, a "golden master," and then that it had been withdrawn from production. The rumors continued: there were over 1200 open bugs yet to be fixed before System 7 could ship; no one in Apple was sure when it would ship; there was even a frantic phone call from one of the Meanies to a committee member saying, "System 7 had been canceled." It was a very confusing first week in May. No one, not even the program manager, had a true handle on what the world was hearing about System 7. There were at least 800 versions of what was happening: one for every member of the System 7 development team and one for every member of the System 7 marketing team. These stories were mutated by the leaks at Apple and the telling and retelling of the stories at User Group meetings. At the Sunday registration for WWDC there were several third-party developers comparing notes about what they had heard about System 7. If the wildest parts of all the tales were strung together, it would have been:

MacWeek stole the disk that System 7 was on and the MacWeek staffer was shot by a security guard. The California Highway Patrol impounded the disk for evidence and the backup system was not working. While the disk was impounded, several companies offered money to the property room manager to lose the disk. Also, there had been a serious earthquake under the Apple campus, and several of the key people had fallen into a crack in the earth so the code could not even be recreated. And finally, Apple ordered that all the printouts were to be burned, to prevent them from falling into the wrong hands.

The reality of the story is much more mundane. The final build (compile) had not passed testing. There was an old version of a piece of code that had been included accidentally in the build. The programmer who had the correct version was off-campus and could not be reached for a couple of hours. The rumors were built from that single incident. That is not to say that there were not a large number of heros during April and into May. The whole story of that 8-week period is lost in the bin of urban legends that it created.

Kirk Loevner was the featured speaker at the first session of WWDC. He began by putting up the same time line that had been used at the 1989 and 1990 WWDC conferences to talk about System 7. Groans came from the back of the room. Everyone expected to hear that System 7

was delayed once again. The session started with Kirk on four of the large screens at the front of the room. Then the images on the screens changed. Apple used the large screens at the front of the hall to show carts moving on a loading dock. They left the loading dock, and the carts moved into and through a nondescript hallway. The room was filled with silence; no one could figure out what was going on. The carts each towed a yellow trailer; some towed two. The carts were making a warning noise—"beep"—"beep" and had flashing yellow lights. The walls were opening at several places around the room as the carts continued to move through hallways. The sound was confusing as if someone had doubled the number of speakers. Someone shouted as the cart passed a temporary sign in the lobby of the convention center. The beeping was getting louder, was it the sound system? A rumble rose from the audience as the confusion intensified. No one was sure what was going on. At the microphone, Kirk Loevner was talking about the history of System 7, but no one was listening. The walls reflected a yellow flashing light, the camera crews appeared in the now opened walls. Then the tractors appeared. The hall was in confusion. When the camera cut to the audience and it appeared on the big screens the place went berserk. Yellow lights were flashing on the carts, and the room was filled with cheers. The first Meanie pulled a shrink-wrapped box with System 7 on the cover out of the cart, and the room dissolved into bedlam. Kirk Loevner tried to give instructions, but was unable to get the volume on the PA loud enough for anyone to hear him over the chaos. For three years the programmers in that room sweated for this day. The chaos returned to an emotional glow after about half an hour. No one timed it—but to some it was a time frozen in history—to others it zipped by in seconds. For many, it was Apple's finest hour.

Many programmers did not attend another session that day; instead, they retired to their rooms to install System 7 and test their programs against it. There was a trickle of people back into the conference all day with comments like "It works, it really works." or "I cannot figure out what they changed, anyone seen C.K.?"

There were tired people coming back after installing the software, their program, and reporting back results. Many were happy fans of System 7, they had followed all the rules and notes, their programs

worked; they were System 7 savvy. The rest were coming back to the hall and looking for members of Developer Technical Support (DTS), like C.K. Haun, for advise on how to make the programs run better or to help identify a bug. The DTS debugging lab did not close on time all week. In many cases when security pushed people out of the DTS lab, the DTS staff member followed the third-party programmer back to the hotel they were staying in. Many people worked around the clock for the whole week, sessions during the day, and debugging at night. There were 5000 very tired people by the end of the week.

At MacHack in 1989 there had been a pool for the date that System 7 would ship. The last date in the pool was September 1990. Apple missed the shipping date for System 7 by over 18 months! The attendees at MacHack were off by over 9 months in their worst nightmares. In the mean time a lot of technology was thrown at the Macintosh Operating System. QuickDrawGX was in its fourth year of development and looked like it would have an impact on users within a year. QuickTime was announced at WWDC in 1991. The alpha version of it was made available to everyone at WWDC in 1991. Many parts of System 7 had changed again since the 1990 MacHack. This caused a number of programmers to rewrite the hacks from 1990 and put them in the contest again in 1991. Because of the overwhelming number of entries the year before, Greg Marriott and Scott Boyd both decided to limit the number of hacks that a single programmer could enter.

There was a mad scramble to deal with QuickTime and the resurgence of QuickDrawGX at MacHack after WWDC. Several sessions for MacHack were decided on the fly at WWDC. The speakers from WWDC for GX and QuickTime agreed on the spot to come to MacHack and lead the sessions. They all indicated that MacHack was something that they wanted to try. There were five weeks to pull the whole program back together.

Tuesday started on a sour note; the network was not coming up in the machine room. MacHack was not getting off to a great start. The techs from the University of Michigan ended up reloading every machine from scratch to get the network up and running. Most of them were doing it on their own time. The university had not authorized overtime to set the machine room up. Several were asked to stay and

attend MacHack, but there was a rumor that the keynote speaker was not coming.

There were four or five programmers with kill Deans INITs buttons on, and Deans INITs were the subject of more than one discussion. Programmers killed time in the hallway outside the machine room waiting for it to open. During the past year, Dean learned a lot about stable programming and it showed in his 1991 entries. System 6 was a thing of the past, as almost all the hacks dealt directly with System 7 at MacHack in 1991. The schedule from the 1991 MacHack was loaded with sessions on System 7 and QuickTime. There was equipment to take advantage of QuickTime and System 7 in the machine room. Tuesday ended very late for most people. The machine room opened several hours late, but it opened and worked as advertised. The University of Michigan and CAEN had come through again.

MacHack 1991 on Wednesday morning started with a session on *debugging*, the favorite topic of most programmers. Debugging is the long and tedious process of walking through a program over and over to find and fix problems. It is debugging that separates the great programs from the average programs. For five hours, four of the best debugging experts on the Macintosh discussed and demonstrated debugging techniques that make efficient use of a programmer's time. Code was offered by the audience in addition to the code that was brought by the panelist to illustrate points. The session ran through lunch and did not really end until Sunday morning, when the last machine was removed from the hotel and loaded into a car. If it had run on batteries, the debugging session would have continued all the way to the airport. This session was so successful that it has been repeated every year since. After lunch was the keynote. Several people had seen Kirk Loevener, the director of the Apple Developer Group, enter the hotel. Kirk was to be the keynote speaker. Six weeks earlier, Kirk was a hero, and the choice of Kirk as keynote looked like a very good one. It was not looking like such a smart move now. There were little discussions at every table about what was and was not going to happen at the keynote address. Apple had made some moves since WWDC that had not gone over well with the third-party development community. Kirk Loevner had gone from hero to goat in 6 weeks; System 7 was old news, regarding Apple's developer programs.

Kirk stood up and in his best Harvard MBA way started to discuss the market share of the Macintosh, complete with slides that included the tag line "Apple Marketing." His keynote was about how important third-party developers were to Apple and how Apple was refocusing on these third-party developers. It was a marketing pitch from the first slide given to a technical audience. Kirk droned on, the developers stood up, wondered to the bathroom, clustered in the hall, doodled, and even started competing conversations. There was a light smattering of applause when Kirk finished. Kirk walked into a room full of loaded guns and proceeded to ignore the problem, but it did not stay that way. Once question and answer started, the gloves came off. The first question was not about market share, but rather, the doubling of the cost of the developer program. The second was on the increased difficulty in becoming a developer. The third question dealt with the changes in the evangelism program. The questions grew tougher and tougher. Kirk acknowledged these problems and went on to admit that Developer Program was now a profit center for Apple with its own profit and loss statement. This was a major change in the way that Apple viewed developer support in the past; Apple looked at developer support as a marketing expense. They used the program to foster companies like Aldus and ACIUS. The cost, they reasoned, was worth it. The applications sold the machines, not the other way around. Kirk stated further that Apple needed the developer programs to make money directly in order for Apple to continue to work closely with developers.

Many of the attendees felt that the statement that Apple really needed third-party developers and that the developer programs needed to make money for Apple to keep Apple committed were contradictory. Several of the more cynical attendees felt Apple needed third-party developers in order to make sure that the company was profitable, through the fees the developers paid. After the session ended, Kirk retreated to his room with several of the long-term Apple attendees.

A new feature of the 1991 MacHack was the code clinics. Veteran programmers, like Leonard Rosenthal, would sit down one on one with a new programmer and review code, assist with a problem, or discuss a part of Macintosh programming that the programmer was having trouble with. Bill Fernandez, the first Human Interface Specialist to work on the Macintosh interface, spent an afternoon one on one with

programmers. He discussed the finer points of the interface to their programs. These sessions were presented Wednesday afternoon and were the calm spot at MacHack. The bash Apple session was yet to come.

Kirk stayed for the Bash Apple session that evening and again found himself at the center of a fire storm. With the changes in pricing on developer programs, evangilisism moving to support large developers, and tightened requirements for being a developer; people were angry. It felt like Apple was cutting its roots to many folks. Both Jordan Mattson and Sheila Brady were on the defensive. It was a very long and angry session, harking back to the first bash Apple session. If it had not been for Darin steering questions back to technical issues and Jordan all but dancing on the table, Kirk might have been lynched. When the movie *Terminator 2* was released, Kirk was given the nickname T-1000 by the third-party developer community. Kirk left early Thursday morning and with him the Apple executive in residence feeling left. After he was gone, the mood became more upbeat. The Bash Apple session had again allowed everyone to vent, and the attendees could concentrate on writing code again.

Waldemar Horwat again rounded up a set of stunning papers for the paper track. They included topics like Multiplatform software development, Dynamic Applications on the Macintosh, software testing on a shoestring, and languages for scripting. These were topics that the rest of the industry did not start to discuss for another year. Again the paper track was building a bridge to the future and opening doors early for programmers who wanted to live on the edge. The Dynamic Applications paper was a great overview of some of what Apple's OpenDoc project was to be about. The Languages paper dealt with languages like AppleScript, which shipped two years later. In all, the papers stimulated discussion and provided a forum for new ideas.

The balance of the Apple folks were hackers like the rest of the attendees—not policy makers. Sessions on Object Oriented Programming, QuickTime, MacApp 3, the Menu and Window managers were all heavily attended. The hands-down favorite session was "Hacking, Patching, INITing, and Crashing," done by the veteran hackers Scott Boyd, Chris Derossi, Dean Yu, and Robert Herrell. It was a rousing good time with a mix of war stories and practical tips. Dean was in great form and took many barbed comments from the audience. Dean

was no longer a student; he was now a member of Apple's system software team which was a clear admission of how much Dean had learned in the previous year. Always a brilliant coder, Dean had matured into a brilliant and savvy programmer. His hacks were debugged, able to run with other extensions, and offered documentation that was readable and useable. By the end of the week, everyone was leaving the "Kill Dean's INITs" buttons in their rooms.

After the banquet, Thursday night, everyone returned to the hotel. The outdoor banquet had been cut short by a thunderstorm. The Hack contest, scheduled for midnight, started late. There were plenty of laughs, the crowd favorites were OKOKOK, TooManyLawyers, and Makin' Copies. At the hack contest, each presenter was allowed free reign. This lead to a number of comic routines, several of them to hide problems with loading software or crashing. OKOKOK won an ovation for removing from the realm of annoying to the realm of trivial dialog boxes that just seemed to pop up without reason. Forgetting to clear a dialog box when the printer is up one floor and across the building is a real pain. Makin' Copies was cheered, it made waiting for disks to fill a little less hostile. TooManyLawyers brought wild applause; the name said it all.

Sheila Brady, the manager of the System 7 project, spoke on Friday afternoon. People who had spent the night in the machine room left wake up calls for Sheila's session. Most of the non-Apple folks knew Sheila only by reputation, which did not jibe with her external appearance. Sheila is not your typical project manager. No one outside Apple understands the intensity that Sheila is capable of bringing to a project, but many people learned that there was iron behind Sheila's words that Friday afternoon. Several had watched her work to finish a hack for the hack contest and were amazed at the detail and tenacity that she put into the process. The session was mobbed. The committee moved two other sessions in order to make room for Sheila's presentation. Sheila discussed candidly what had gone right and wrong in the System 7 process and how Apple had put over 1000 effort years into the creation of System 7 with over 700 people working on the project

at the peak time. It was the largest software engineering project undertaken outside a government contract to date. Rules about clear specifications, freezing Application Program Interfaces (API) early and bite sizing the work, so that a team of two or three people could complete the task in a short period of time were key points to Sheila's talk. Sheila also laid out in extreme detail the history of Big Bang, the code name for System 7. This was the first time that Apple had talked in an open forum about what had happened during the System 7 gestation. There was a feeling that Sheila was letting everyone in on the family secrets.

The time line Sheila used was:

- 4/88—Greg Marriott joins Apple.
- 7/88—Big Bang!—Apple formally starts System 7 work.
- 11/88—Darin Adler becomes the scapegoat of Big Bang. Darin is name the lead integrator for System 7.
- 12/88—New 7.0 Testing Management.
- 1/89—Chaminade Offsite—First Blue Books. This is the first real specification for System 7, about a year into the initial project.
- 3/20/89—Chris DeRossi, from Developer Technical Support Joins the Blue Meanies.
- 3/21/89—The Blue Meanies, the software integration team, are formed. Darin Adler is the lead programmer and group leader, the group will grow to 13 members before System 7 ships.
- 4/89—Begin the Alpha build that will ultimately be released to developers at WWDC in May 1990.
- 5/89—System 7 is announced at Developers' Conference.
- 6/89—Chris DeRossi wins the Hack contest at MacHack with Color Finder, which found its way into System 7.

- 6/89—Schedule for 7.0 pieces is confirmed, now there is a schedule that everyone is working to and there are real time lines for product shipment.
- 7/89—QuickDraw GX, the new print architecture and the line layout manager are pulled from System 7.
- 8/89—New 7.0 Testing Management, for the second time, since the project started.
- 8/89—Rich Castro is put in charge of System 7 development.
- 8/89—Balloon Help and Apple FileShare are added to System 7's core functions.
- 10/89—Management of the System 7 development team changes.
- 10/17/89—Earthquake!All of the System 7 engineers are forced out of the building that they have been working in. Over 800 people at Apple have to be relocated. For two weeks no real work is done on the software. When the move is over, the System 7 team is scattered between six buildings.
- 11/15/89—Official Alpha testing starts on System 7, 16 months after the project is started.
- 12/89—Apple reorganizes the whole company and again changes the testing management for System 7.
- 12/89—Sheila Brady is named manager of the System 7 project.
- 2/90—John Louis Gassee announces to the world that he is burned out and is leaving Apple.
- 3/90—New 7.0 Testing Management.
- 4/90—Greg Marriott rejoins Apple Computer, after burning out and leaving for a few weeks.

- 5/90—System 7 is announced at WWDC, the Alpha CD is passed out to developers two years after the project is started.
- 6/90—New 7.0 Testing Management.
- 6/90—Dean Yu wins the Hack contest with Net Bunny.
- 10/90—System 7 is officially moved into the Beta testing stage.
- 1/91—Dean Yu drops out of the University of Michigan to join the Blue Meanies.
- 2/91—New 7.0 Testing Management.
- 5/13/91—System 7 is announced at Developers' Conference and is shipped over two years after Apple first promises it.

Sheila talked about the confusion that changing test management five times and project management four times had on the process. She also went into detail about how confusing the project was at times, as people and whole organizations either did or did not exist on the Apple organizational charts during the reorganizations. At one point the Blue Meanies did not exist in the Apple organizational charts after a reorganization for several days. Sheila was candid about the parts of System 7 that the development team was not thrilled about shipping, including the The Chooser, the fact that Get Info comments can still be lost, and that the disk icons are not in color; all minor items. Sheila related the build process (see Figure 3.1) that Apple may have used for creating System 7. Sheila also talked about a deck of note cards, which would contain ideas that could not go in System 7. These note cards were pink, hence the name for the new version of Apple's operating system "Pink," which was the basis of Taligent's first operating system. Two quotes from Sheila have appeared over and over in the press: (1) "You are empowered by yourself first, your team second, and finally, by management." and (2) "The schedule that shipped 7.0 was the right one. Too bad it took us so long to figure it out!"

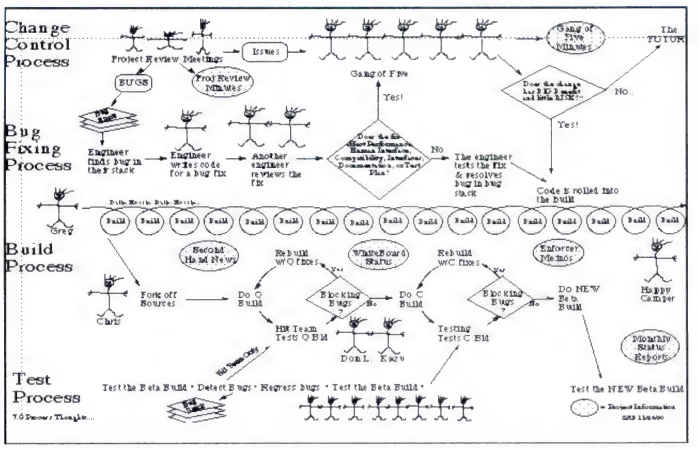


FIGURE 3.1 SHEILA'S UNDERSTANDING OF THE SYSTEM 7 BUILD PROCESS

The session was a roller coaster, with Members of the Meanies adding comments and asides throughout the session. On at least one occasion, Sheila told one of the persons making comments that they could continue outside, which brought laughs and hisses. The audience clapped, applauded wildly, and laughed. In short, it was the session that everyone hoped for. Sheila almost single-handedly restored the faith in Apple in that two-hour session. Almost everyone there said that Sheila should have given the keynote. In fact, Sheila so impressed Apple with the end result from System 7 that they named her manager of the Power Macintosh Project, "Rock and Roll".

Sheila was one of the growing number of women at MacHack. In 1986, MacHack had a number of academic women involved. By 1988, the conference was almost all male. In 1991, MacHack was over 10% women, and that trend has continued. Women as programmers are rare. There have been a number of studies on why this is the case, and they all fall back on the idea that women do not view computers as fun toys like young men do. Rather, women think of computers as tools to get jobs done. The fact that MacHack was over 10% women and that they all programmed was an interesting situation that has been explored in a number of MacHack discussions. Most of the women involved have a story about how they came to enjoy computers. In many cases the women's stories mirror their male programmer's stories. In

short, once a bright young person is interested in computers, gender becomes irrelevant.

The hack awards on Friday were anticlimactic; most of the winners were expected the favorites from the audience were all the real winners. Most of the winners were team efforts, most of which included both Apple DTS and nonapple programmers. The conference ended with less tension and a feeling that the people who counted, DTS and the actual Apple programmers, still were on the side of the third-party developers. As MacHack was ending, a number of programmers who were key to the development of System 7 admitted that they were leaving Apple. Darin Adler, the team leader, left for General Magic and David Feldman for Specular International to create Infini-D. Over the coming year, more than half of the Blue Meanies, would leave for other companies, taking with them knowledge of how the operating system really worked. This loss of programming talent would slow the process of getting the next version of the Macintosh Operating System out by over a year.

1991 was the year that turned the corner for MacHack, no longer was it just an annual conference, it was a family that had members inside and outside Apple; a family with one purpose, that is, great software, killer applications!

QuickDrawGX, QuickTime, and AppleEvents

Three key pieces of the future were under development at Apple in 1991, all of which had an impact on the hacks that were created at MacHack and have had or will have an impact on the operating system for the Macintosh. QuickDrawGX, was a massive project to rewrite most of the graphics routines that were burned into the original Macintosh ROMs. It was a project that had seen both good and bad times. Originally scheduled to be part of System 6, it slipped to System 7 and then outward. As this book is being written, QuickDrawGX, in beta for over a year, might ship as part of System 7.5. The project has had several changes to the specification and a number of backward compatibility requirements dumped on it. GX started out as a simple update to Quickdraw; it began as a simple RGB color standard, then

postscript was added and removed, True Image was added, and then postscript was added back in. Level one postscript was replaced by level two, and finally, it went from 2-D to 3-D. GX was constantly changing. In fact many of the requirements that GX carries with it forced the long development and beta test period. Because it is replacing a part of the current operating system, GX is the subject of intense testing and lobbying pressure. One camp lobbied for postscript, another for true type. One camp wanted 3-D, another did not. One wanted every routine to be as fast as possible, another wanted the software to just ship. GX, a unique Macintosh feature, is the heart of the graphic user interface. Because of these fast graphic routines, which are easy to use by programmers, the Macintosh has maintained the lead in the way the interface looks. But GX is a massive undertaking, with more code involved than there was in all of System 7. Also, because the project has continued to slip, the machines that it is aimed at continue to grow in complexity. GX was intended for a 68020, the original Macintosh II, and with the release of the Power Macintosh, Apple will have changed the hardware under GX for the third time. Color QuickDraw and 32-bit QuickDraw were both intended as part of the original GX project, but because the whole project was not complete, these pieces were shipped separately. This means that the GX code base had to take into account additional backward compatibility. QuickDrawGX is a project that will offer advantages in speed and programming ease and whose impact will be felt for years.

In contrast, the QuickTime team created a whole new functionality for the Macintosh. There were no issues about installed base and backward compatibility because there was no base and nothing to be compatible with. In the operating system, there were no calls to display video, so the QuickTime team had an open field to do the work in. Even though the QuickTime software was over 20% of the size of the GX code, the number of programmers and testers assigned never approached 20% of the GX team. QuickTime was a two-year project from conception to shipment. It offered a unique new functionality to the Macintosh, one that forced Microsoft to counter with Video for Windows. Apple even shipped a version of QuickTime for Windows within a few months of the release Video for Windows. QuickTime is rapidly becoming a multiplatform standard. The good news is that at MacHack, the QuickTime team explained the lessons they had learned

from the QuickDraw project and how the QuickTime software was designed to grow with the capabilities of the Macintosh. Unlike GX, where every routine is part of the whole, QuickTime takes advantage of modular programs, so as the compression software is upgraded, it just plugs and plays, instead of having to be completely rewritten. In fact, Apple has shipped three different compression schemes in three different versions of QuickTime. In the three years since QuickTime's release, the software has undergone three major revisions, resulting in version 2.0, which is in the hands of key third-party developers.

AppleEvents was a whole different effort. Where QuickTime had an open field and GX required backward compatibility, AppleEvents required competing companies to cooperate with each other. The function of MacHack has been to get people on the programming level together to hammer out standards for AppleEvents. The virus and utility suites of events were both worked out in the evening at the 1991 MacHack. Progress was made on the Spreadsheet, telecommunications, and word processing suites during those same evenings. Though MacHack has never officially taken a part in making alliances, it is surprising what a group of programmers, who are sharing a common platform, are willing to discuss when the marketing and management are not looking over their shoulders. During the discussions, there were even trades of Application Programming Interface (API) specifications, which made other programmers aware of the problems that one set of calls would cause. Real issues with real code were delt with in an open and honest exchange.

Brita Meng from MIT to *MacWorld* to Shiva

Brita Meng, is a two-time chair of MacHack. As a student, she excelled in both hardware and software and began to write about computers as she worked toward her Bachelor of Science degree in electrical engineering at MIT. The editors at IDG, the parent of *MacWorld*, saw several of her articles in the Boston Computer Society newsletters and invited her to come to work for them. Brita worked for IDG from its Boston offices, forming the core of the east coast editorial staff for *MacWorld*. Her efforts in the first year at *MacWorld*,

won her the respect of most of the programmers in the Macintosh community. It was not uncommon to hear a programmer defame an article in *MacWorld*, only to take it all back upon hearing that Brita was the author. Brita also became a speaker on the *MacWorld*, COMDEX circuit. Her easygoing manner and ability to explain the most complex technology in simple terms made her a winner at both jobs. The first public acknowledgment of this her great sense of humor was an April issue of *MacWorld* with the Geek Chic section, which was one of of practical jokes and pranks.

Brita was and is an avid programmer. After all, making a machine do what it has never done before is exciting. One of the few women to enter the Hack contest, Brita's drive led her to seek a challenge that went beyond just writing about technology. She moved to Shiva as the product manager for several products and ended up becoming the manager of all their software. But being driven is not without its pitfalls, as Brita has on occasion had to choose between her work and the rest of the world. In a few cases, work won, but she has never regretted her choices, even the hard ones. She let work get ahead of her marriage, and it took the ultimate toll.

Brita is slow to offer advice, but her advise and her trademarked "ok honey" are both legendary in the Macintosh community. It was in Brita's honor that a group of MacHack folks gathered to take Brita to Great America on the Sunday of WWDC week. Brita, driven as usual, has missed all three trips, she was still working while her friends were riding roller coasters in her honor. Even people who would not ride the roller coasters went along so they could spend an afternoon with Brita who worked through her own party.

The 1991 Hacks

There were over 50 hacks in the Hack contest at 1991 MacHack. Several of them have either been withdrawn from the market, or have gone on to be commercial products. This is the first set of hacks that were built with the release version of System 7.


WARNING

The Centris and Quadra computers did not exist in 1991 and some of these hacks have problems with these 68040 based machines.

Æ RPC Stub Compiler

7 § SW-MPW

A Remote Procedure Call (RPC) Stub Compiler by Paul Campbell, is a tool for use with MPW. It cannot be used as a standalone. The RPC compiler is used to create AppleEvent packets in compiled form that can be sent from one program to another. These packets can then drive can the program that they are being sent to, making it perform a series of steps and returning the answer to the original program. The RPC stub compiler reads a description of a procedural interface (like a subroutine call) and produces the necessary code so that the call may be made to a subroutine in a remote process. In this case the transport used is AppleEvents, and the compiler makes all the AppleEvents calls required to round up all the arguments, send them to the remote process, unpack them and call the server subroutine, and return the resulting value (if any) back to the calling routine.

This compiler is based around C, and it accepts a C-like syntax for its subroutines. It produces as output MPW C, a source that can be compiled to produce the resulting AppleEvents glue. Since it uses AppleEvents as a transport, you must be running System 7.0 or later. To Install it, simply drag the MPW Tool rpc into your MPW Tools directory.

AKA

7

AKA by Fred Monroe is a logical extension of a Finder interface feature. By holding down the **Option** key when dragging an icon from one folder to another, the Finder is forced to make a copy of that file in the new folder. With AKA, aliases are created in much the same way. By holding down the **Control** key when dragging an icon from one folder to another, an alias is created in the new folder, instead of the whole file being copied. This option has been incorporated into the Appearance Manager that is in System 8, which is scheduled to ship in late 1995.

AliasThis!

7

Alias This! by Bruce Oberg and Gordon Sheridan is a *droplet*, which is a program that can only be run by dragging an item on it and dropping it. There was no name for a droplet in 1991, but Bruce and Gordon created one. AppleScript formalized the name droplet in late 1993. Bruce Oberg was working at Microsoft and Gordan in DTS at Apple when this hack was created. It was the height of the Apple—Microsoft copyright lawsuit. The suggested name for the hack was "glasnost," but Alias This! was felt to be more descriptive. It took just over two hours to write and was entered with 15 seconds to spare in the hack contest. Alias This! will create an alias of any file that you drag and drop on it, in your Apple Menu. This saves three steps over the normal method of creating aliases and placing them in the Apple menu.

AniMicons

7 § SW—HyperCard 2.x

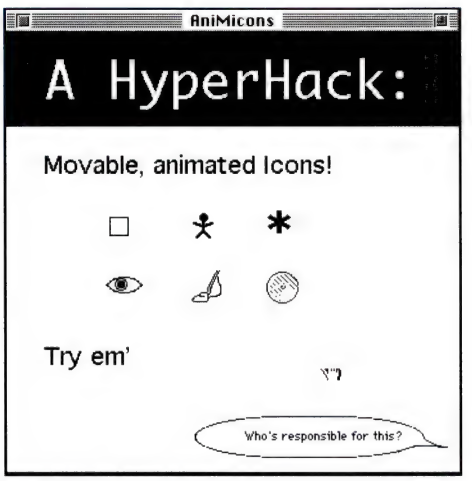


FIGURE 3.2 ANIMATED ICONS IN THE ORIGINAL POSITIONS

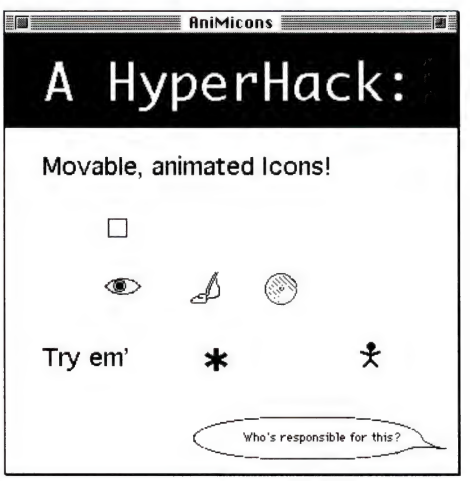


FIGURE 3.3 ANIMATED ICONS AFTER BEING PLAYED WITH

AniMicons, by David Drucker of BCS, is a HyperCard Hack. Using a series of simple icons, David made each icon animate itself when the cursor was moved over the icon and the mouse button held down. Additionally these icons can be moved all over the card (see figures 3.2 and 3.3). More complex icons can be created to allow simple games or silly stacks to be created. All of the hard work was done in HyperCard and the scripts are available to read and copy. There are non-trivial animation scripts and can be used with more than just simple icons. HyperCard 2.x is required to make use of this hack. It is fully compatible with HyperCard 2.2 and coloring the icons is a very fast process.

Ashtray

7 §

Ashtray by Kevin McDonell is a system extension that converts text into PigLatin. In its current implementation, you have to hold the **Option** key down to get the feature to work.

Ashtray can be fun when there is a high pressure deadline and no one will lighten up. Just make a copy of a critical piece of the presentation and offer it to the most serious person in the group. Always work from a backup version of the document.

Basura

7

Basura by Fred Monroe is another of his logical extensions of the interface. Again Fred is trying to keep the user's hands on the keyboard. In this case it is for deleting files, which is one of the few Finder functions that has to be done with the mouse; there is no keyboard shortcut. Rather than using the mouse to throw the file in the trash, Fred's hack allows users to do it by highlighting the icon(s) and then

choosing the **Command** and **Delete** keys from the keyboard at the same time. This combination throws the item into the trash, where it can later be deleted from the disk.

Bully

7 §

Bully by Scott T. Boyd kills the Finder in System 7 when an application is open. This makes a 2-megabyte (M) Macintosh useful. To use Bully, which is a droplet, Quit everything on the machine and then drop an application on Bully. Bully will force the Finder to quit and return approximately 500 kilobytes (K) of memory to the application that was dropped on it. This is the amount of memory that the Finder needs to run. With the exception of the Finder, Bully will not cause any application that is running to quit. On a 2-M machine with System 7 installed, Bully offers 1.2 M of usable memory to the application that is dropped on Bully.

Clarus—the Tail Patch

7 §

Clarus—the Tail Patch by Dave Ewing and Tim Senecal is a small extension to make the **About** box for System 7 more interesting. Clarus is the official name for the DogCow, the black-and-white-spotted animal that appears on most the developer mailings and on all the official developer technical support documentation. A *tail patch* is a small piece of code that is loaded at the end of a program, to modify its behavior. This is opposed to the normal method of patching, called a "head patch," where the patch is loaded first. Normally, extensions are tail patches, because the system is already loaded, or at least significant portions are. The beta versions of System 7 had an incredible **Scrolling About** box with the names and jobs of all the people who were part of

the System 7 team. When the real System 7 shipped, the animated, **Scrolling About** box with music disappeared. Clarus was instituted to bring back some of the fun of the **About** box. Using the original System 1 **About** box as a starting point, the two programmers instituted an animated dialog box that includes Clarus mooing. The names of the programmers involved in creating the Finder are included in the box, whose animation continues for a couple of minutes. It is impossible to convey the fun of this patch in a picture; install it and give it a try.


WARNING

Be aware that Clarus and Cameraman 2.0 do not like each other, and Cameraman will continue to shoot screen shots as long as Clarus is installed.

ColorHack 1991

7 § SW-MPW

ColorHack by Ray Sanders and Steve Antonakes is a patch to MPW that will make the reserved words, the commands of the language, appear in color on a color screen. This is very useful when tracing source code and determining whether something that was written may have an extra command in it or may be missing a command. When a programmer has been debugging for several hours, having reserve words in color can be a life saver. Sometimes a line of code gets marked as a comment, when it should not. The program misbehaves, and the problem can be very hard to find. With the reserved words in color, it becomes very obvious, that the line should be active code and not a comment. In order to work, it requires MPW and an MPW compiler. But all of the source is included, which makes porting it to another compiler a fairly simple task.

CommanderTabs

7 §

CommanderTabs by Sean Parent is a quick hack that makes tabs work in TextEdit documents. Normally, there are no tabs in a TeachText or other TextEdit document, but Sean added Tabs to the document. Since the **Tab** key is used in TextEdit to change from screen to screen of text, using **Command-Tab** as a key combination switches between moving through the document and making tabs in the text. This is where the name CommanderTabs came from.

CoolLW

7

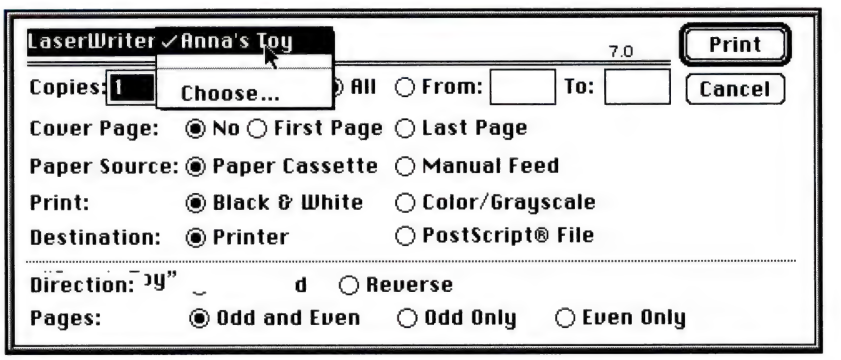


FIGURE 3.4 COOLLW WITH THE MENU ACTIVATED

CoolLW by Byron Hand and Leonard Rosenthal. CoolLW removes most of the reason to go to the Chooser. CoolLW attaches a menu to the print dialog that is activated every time Print is chosen from the menu (see Figure 3.4). This menu contains all the printers in all the zones that the CoolLW can see. This is a very long menu on a large university network. The exact printer for a document to print from can be chosen from the menu, without having to resort to the Chooser. Each document can be printed to its own printer. The Print Monitor will track which job is sent to a printer and spool the appropriate file to it. This is useful when attempting to print complex graphics to a printer and then having a quick invoice to dash off. While the first printer is busy with the first job, the second job can be dispatched to another printer in a very quick and painless fashion. CoolLW also reminds the printer which user was last used. It is listed as the top menu choice in the menu. Because CoolLW is based on a System 7.0 laserwriter driver, not all the features of many of the new printers are active when using it. Be aware that the printer will behave like a standard LaserWriter NT when this version of the LaserWriter driver is used. Since the driver file is named "CoolLW," there is no worry when dragging it into the system folder about replacing the regular laser printer drivers that are installed. To use CoolLW, drag it into the system folder and then choose it in the Chooser. There is no need to even restart the Macintosh.

DOS sHELL

7 §

DOS sHELL by Jay Wiess scared a number of unsuspecting programmers at MacHack. This simple hack allows the Mac to change sides. It boots the machine as if it were a DOS machine, complete with C prompt. With DOS sHELL loaded, the machine will stop mid-restart and throw up a screen with "MS-DOS 3.3" at the top. Originally, the hack had an **Escape** key that was secret to let people out of the C: prompt. Because of the negative reactions of the various experienced programmers to having a C: prompt on the screen, Jay modified the hack so that the first key the user touched causes the program to disappear and the boot process to continue. (This hack makes a great practical joke.)

Dropple Menu

7

Dropple Menu by Fred Monroe and Steve Falkenberg allows a user to drag a document onto an item in the Apple menu and have the item in the Apple menu operate on the document like it would if the document were dragged and dropped on it in the Finder. For instance, when MacWrite is installed in the Apple menu, dragging a text document onto the Apple menu and then down to MacWrite will cause MacWrite to open the text document, as if you had opened MacWrite first, and then chosen the document with the **Open** command in the file menu of MacWrite. When there are two or more word processors or drawing programs installed on a disk, this can be useful in making sure the correct word processor opens the document, rather than getting, "The application that created this document cannot be found, would you like to open it with TeachText?"

DropSave

7

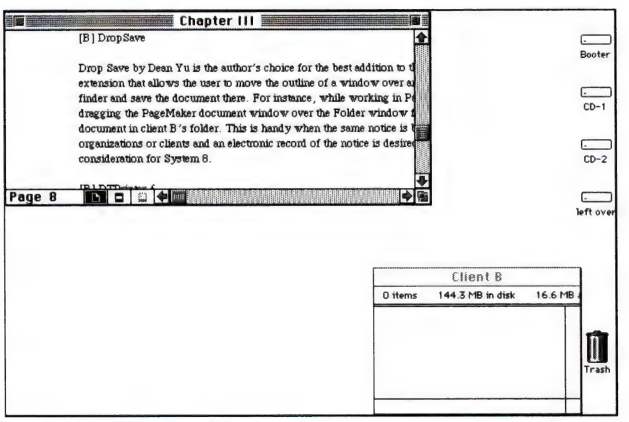


FIGURE 3.5 THE DOCUMENT AND THE **TARGET** FOLDER, BOTH ON THE DISKTOP

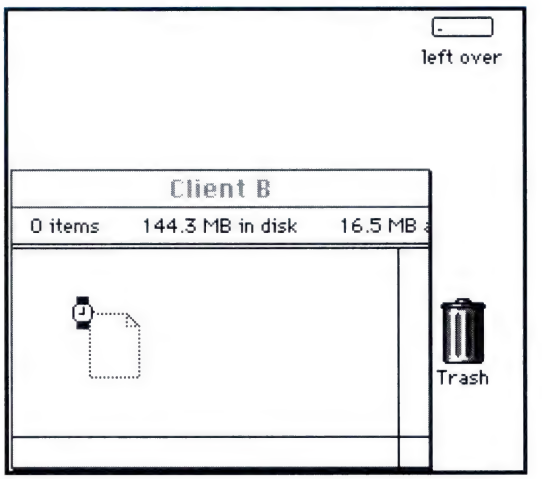


FIGURE 3.6 THE DOCUMENT WINDOW HAS CHANGED TO A DOCUMENT ICON

Drop Save by Dean Yu is this author's choice for the best addition to the interface. Drop Save is an extension that allows the user to move the outline of a window over an open folder window in the finder and to save the document there. For instance, while working in PageMaker for Client B, dragging the PageMaker document window over the Folder window for Client B saves the document in Client B's folder (see Figures 3.5 and 3.6). This is handy when the same notice is being sent to several organizations or clients and an electronic record of the notice is desired.

To use Drop Save, hold down the **Shift** key while clicking in the title bar of the window. Once the title bar is clicked, the window shrinks down to a document icon. The icon will remain as long as the mouse button is down. The mouse can then be used to drag the document icon to the Finder window that the document should be saved in. Drop Save is very touchy. If the wrong extensions are loaded with it, it will work up until the document should be saved. Once the document has been dropped over the window, an error 48 will occur and the document will not be saved (however, the window for the docu-

ment will reappear and the document can be saved normally). This can be cured by removing extensions until it does not happen anymore. Once the document is saved, the window will reappear in the same location it started in. Multiple drags to the same window will work the same way that using save in the menu works in updating the saved copy. Drop Save is under consideration for inclusion in System 8.

DTPrinter

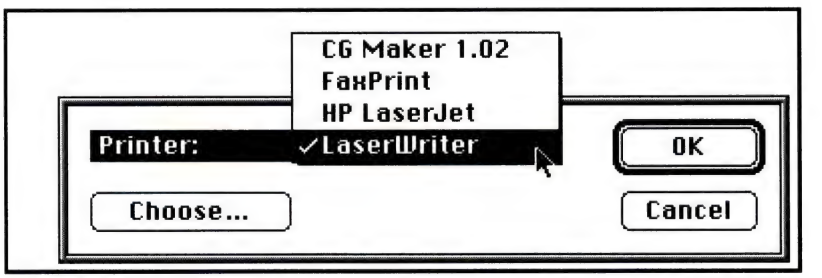


FIGURE 3.7 THE PRINTER SELECTION CHOICES

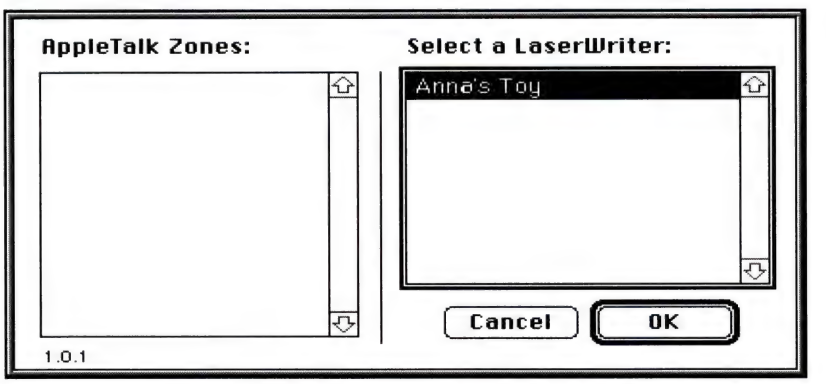


FIGURE 3.8 SELECTING A NETWORK PRINTER

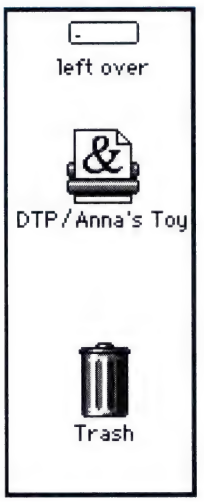


FIGURE 3.9 THE DTPRINTER ICON IN PLACE AND READY TO GO

DTPrinter by Leonard Rosenthal places icons on the desktop for each printer that a user might choose. In 1988, when Apple first talked about System 7 to programmers, it included both QuickDrawGX and the New Print Architecture. There was a new Print Architecture that included a feature like this, called Desktop Printers. This feature placed an icon on the desktop for each printer on the network. The user could trash any icon that they knew they would not use. The New Print Architecture was one of the features dropped from System 7. It was shifted from being part of System 7 to being part of QuickDrawGX. Leonard thought that the Desktop Printers were very cool, so he decided to create them for System 7. Each is configured separately. In order to configure the DTPrinter, take the following steps:

- Make a copy of the DTPrinter application—This is very important, because the customization process is not reversible.
- Double-click the DTPrinter application, so that it will launch and its menu bar will be available.
- Choose **Customize** from the File Menu.
- Select the Type of printer (see Figure 3.7) and the printer name (see Figure 3.8).

- The application will then rename itself DTP/"name of the printer" (see Figure 3.9).
- Drag the icon from the **DTPrinter** folder to the desktop.
- To use it, simply drag and drop the document icon onto the printer icon and a Print dialog box will appear.


NOTE

DTPrinter uses the printer drivers installed in your system folder; therefore, as the drivers are updated, so is DTPrinter. It can access all the features of any printer, which is in direct contrast to the limits caused by modifying the printer driver directly.

Finder Hack

7

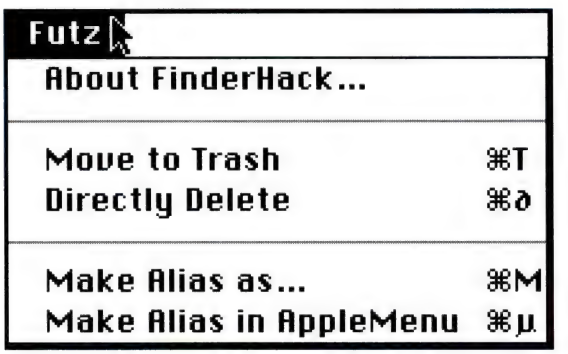


FIGURE 3.10 THE MENU FOR FINDER HACK ACTIVATED

Finder Hack is by Donald Brown, the author of QuickKeys and MockPackage for CE Software. Don has extensive knowledge on how the system actually works on the Macintosh. Finder Hack takes advantage of this knowledge base to make a number of short cuts. This extension places a menu in the menu bar of the Finder (see Figure 3.10) to allow users fast access to a number of features. First, FinderHack is able to move a selected group of icons from their window into the trash, or if preferred, it can delete them without ever moving them to the trash.

Both operations require some bizarre manipulation of the operating system. Next FinderHack is able to make an alias of a file or group of files in the folder they are currently in or in a folder specified by the user. Finally, it is able to make aliases of files and put them directly into the Apple menu. All of these operations save steps and time. Finder Hack is very stable and easy to use. It does, however, rely on the way the system works now. There is no telling when it will just stop working. Many of the features in Finder Hack are in the Appearance Manager, which will be part of System 8.

Fuzzy Balls

7 §

Fuzzy Balls by Jon Wind is a screen saver. Jon took a standard After Dark screen saver module that he had written and converted it to an **FKEY**. This standalone screen saver draws random-sized balls on the screen in color. The balls range in size from half an inch to an inch. The screen saver can be started with a click of the correct function key. It turns off with any key, disk insertion, or mouse button click. The screen saver draws the balls as a series of circles, and one or two balls are drawn each second. This screen saver is handy because until turned on, it takes no memory, unlike After Dark. It is also handy because a single keystroke activates it. This is better than having to move the mouse when the boss arrives unannounced. The **FKEY** is set to **F9** and needs to be installed with **ResEdit**. Instructions for using **ResEdit** are available with **ResEdit** from APDA. The full source code for Fuzzy Balls in Pascal is included.

Help for Hier

7

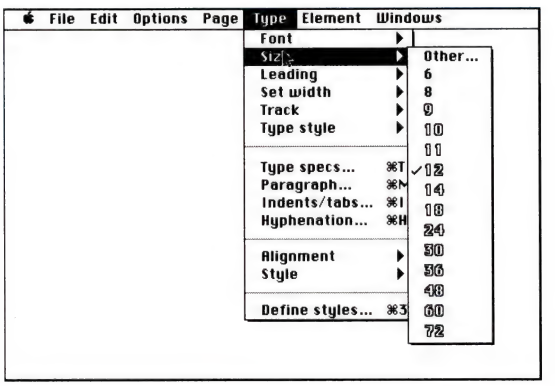


FIGURE 3.11 THE PAGEMAKER TYPE MENU OPEN WITH A SUBMENU

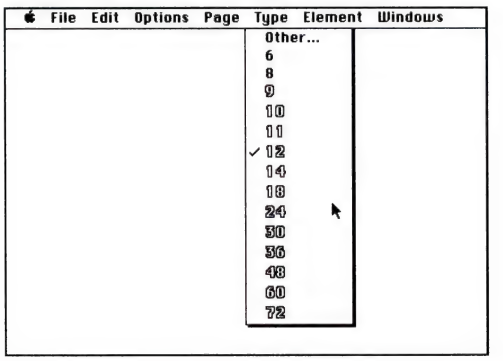


FIGURE 3.12 THE PAGEMAKER TYPE SUBMENU FOR FONT SIZES REMAINS OPEN

Help for Heir by Jon Kalb is an extension that assists new users in using Hierarchical menus. The extension replaces the main menu with the submenu (as shown in Figures 3.11 and 3.12). This allows a user to make full use of the submenu without having to move the mouse into the submenu's normal location or worse, pop up another submenu by accident. Help for Heir is extremely useful in PageMaker and CAD programs, where submenus are the norm, rather than the exception.

It's Just a Clock

76

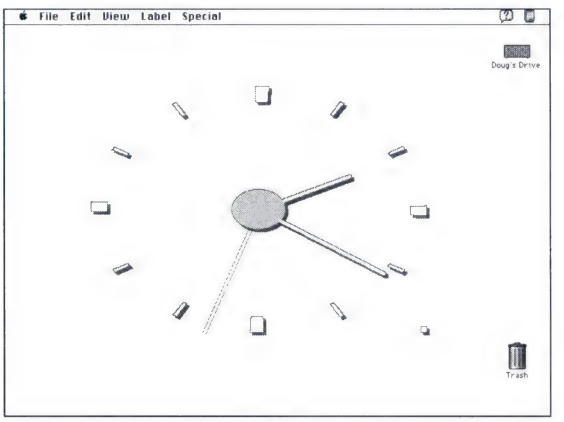


FIGURE 3.13 IT'S JUST A CLOCK, IN ANALOG MODE

It's Just a Clock by Pete Helme of DTS is an application rather than an extension. The advantage of an application is that when it is not running, it takes no memory and it patches no traps, so that it stays compatible with the system software for a longer period of time. The clock is fully configurable, it will use any font in the system folder

and any point size. It automatically resizes when new fonts are chosen. The clock can also be set up as an analog clock (see Figure 3.11), which can be as large or as small as the user chooses. Stretch it, distort it, have fun with it; Pete intended for the clock to be fun to play with.

Kilroy

7

Kilroy by David Koziol is a takeoff on the old Kilroy graffiti from World War II and the early fifties. When Kilroy is running, he will appear at the top of the window, with his nose over the title bar. The two eyes and the nose of Kilroy appear after a preset amount of inactivity on the Macintosh. The user is able to choose the number of minutes to wait before Kilroy appears on the front window. Kilroy is elusive; touch any key, jiggle the mouse, or just breath on the computer and Kilroy will run away and hide. Kilroy and screen savers do not enjoy each other's company. Most of the time, Kilroy just stays home.

MacHack Weather

7 SW-QuickTime

MacHack Weather is the first set of QuickTime movies created at MacHack. The movies deal with the satellite pictures from a weather satellite during MacHack. There is Small Weather and Big Weather. Small Weather uses the compression scheme that Apple shipped with QuickTime 1.0, and it will play with any version of QuickTime. Big Weather is a larger window that shows where the QuickTime team was trying to head with the software in 1991. Big Weather will play with QuickTime 1.6.1.

MacsBugTool

7 § SW-MPW

MacsBugTool by Tom Lippincott is a tool for use with MPW. MacsBug, the Apple-developed debugger, allows programmer to quickly see what is going on in the program that has just been written and assists in determining why a program has crashed. This tool and the script that is included with it makes use of MacsBug from MPW. Normally, to use MacsBug, the programmer's key needs to be pressed. The applications that are running are hidden, and a full screen window for MacsBug appears. This is not very useful when a programmer is stepping through program code trying to locate the exact instruction that causes the machine to crash. MacsBug Tool allows MacsBug to talk to MPW and to execute any commands that the programmer might want to send from MPW. All the communication to and from MacsBug is done in a single MPW window. This is very handy when stepping through a source code for a program looking for the event that triggers the bug. Also, it is extremely hard to save MacsBug windows for future reference, they cannot be printed at all. MacsBug Tool solves this problem, by using MPW's built-in **Save** and **Print** functions. Full source code is included, allowing an experienced programmer to move the tool to Symantec or Metrowerks compilers.

Makin' Copies

7 ~~040~~

Makin' Copies by Eric J. Hayes and Jim Wolff is a talking hack. Whenever the Macintosh begins to copy a file, whether by dragging from one volume to another, or because the duplicate command is selected from the file menu, Makin' Copies announces that the machine is "Makin Copies." This hack arose when the various programmers were copying tools and files over the network to set up the machines that they were programming on. The overloaded network was busy copying thousands of files. Eric and Jim were annoyed with the amount

of time the copies were taking. This lead to a discussion of how to tell how many people were copying files at one time and then to this hack, Makin Copies.

Makin' Copies makes permanent changes to the System file; test with a backup copy of the System file.

MountAlias

7

Mount Alias by Jeff Miller is a program that will mount AppleShare volumes automatically. Mount Alias starts by assigning a folder where all the alias information for AppleShare volumes will be kept on your machine. Then as each AppleShare volume is mounted manually from the Chooser, Mount Alias creates an Alias for the volume in the folder. This alias includes all the password and other information to automatically mount the AppleShare volume in the future. There is a security issue involved, since the password is being stored. In the future, to mount an Appleshare volume, the folder is opened, the icon for the alias of the volume is double-clicked, and the volume is mounted on the desktop. This saves the user many steps. Jeff Miller built this while doing software testing at Apple.

Move Around!

7 § SW-HyperCard 2.x

Move Around by Chris Allen is a HyperCard hack. It takes advantage of Chris's knowledge of HyperCard's underlying mechanics to create two scrolling lists. Items may be dragged from one list to the other. They can be removed by clicking on the item and then the trash can. This allows people to build an intelligent system for building paths through material. Several HyperCard products have taken advantage

of Move Around to provide teachers with a way to customize the lessons for students. This is also useful in building movie lists to play from HyperCard 2.2 or higher.

MS Works Merge Enhancer

7 § SW-Microsoft Works version 1.2 or lower

MS Works Merge Enhancer by Jeff Mandel is an attempt to fix a problem that Microsoft would not fix for several years. Mail merge in the early versions of MS Works was painfully slow. Works had the integrated software field to itself for several years, and so Microsoft would not invest in improving the features. Jeff Mandel had several mail merge projects that needed to be completed in a reasonable amount of time. His hack was an attempt to solve this problem. Today the issue is a moot point; Microsoft has improved the **Mail Merge** feature. But the source code to this hack provides a number of insights on using postscript and building HyperCard XCMDs with C++.

NetBunny 2½

7.0.0 040

NetBunny 2½ by Dean Yu, a version of the Bunny, is solid programming. It is a parody of not only the Energizer Bunny, but of the movie *Naked Gun 2½*. To use NetBunny, install the BunnyINIT in the system folders of the target machines and restart. Then start the rabbit running by double-clicking StartWabbit on one of the machines. NetBunny will be off and running.

It is recommended that NetBunny be uninstalled after use. It can cause crashes on some machines.

NextPrev

7

NextPrev by Alex Rosenberg is a system extension that allows users to move between windows and applications without resorting to the mouse. Using key combinations, a user can move between the windows that are open in a single application, like MacWrite, or they can move between applications. This allows the user to quickly move from document to document cutting and pasting information between documents or referring to one document for information that is to be used in another document.


WARNING

Not all applications number windows in the order that the user opened them, so in some applications moving between windows is a bit like voodoo; the outcome is never certain.

OkOkOk

7

OkOkOk by Dave Falkenburg is a productivity-enhancing utility. When manually feeding the laser printer, forgetting to clear all the dialog boxes before running to the printer to insert the paper can sometimes cost valuable time, not to mention having the boss holding a revised resumé that printed when the manual feed paper was left in place while returning to the computer to clear the dialog boxes. OkOkOk does this job for the user, allowing them to open the print monitor and stroll to the printer, knowing the dialog boxes will be dealt with automatically. The further away the printer is from the desk, the more useful is OkOkOk. This extension will automatically click the default button in a dialog box after 2 seconds. The **Default** button is the one with the heavy line around it and is normally the **OK** button, hence, the name of the hack. OkOkOk is also useful in copying and trashing large numbers of files and building image catalogs.

Primes

7 § SW-TMON

TMON was one of the first debuggers for the Macintosh. It allows programmers complete control over what is happening on the Mac at all times. Primes by Jay Zipnick is a routine for TMON that constantly generates a set of prime numbers in the background and files a block of the Macintosh's memory with them. This routine is useful in testing software on a machine where more than one thing is supposed to be happening at the same time, such as background printing and saving a file or running a screen saver and calculating a spreadsheet. It is especially useful in making sure that all the windows that are drawn on the screen are drawn as complete objects off-screen and then moved to the viewing area on the screen, so that the programs do not look like they have crashed while opening the window or dialog box.

ReturnOpens

7 §

Return Opens by Keith Nemitz offers a simple extension to the system. Select and highlight a file or group of files, then hit the **Return** key to open them from the Finder. This simple hack became a project when Bill Fernandez commented in a Human Interface session that the only difference that a user should see in opening files from the Finder or from an application was that return would open files from the standard Get File dialog box and it would not in the Finder. Now, return works in both cases.

Scott's Analog Clock by Scott Schmitz

7

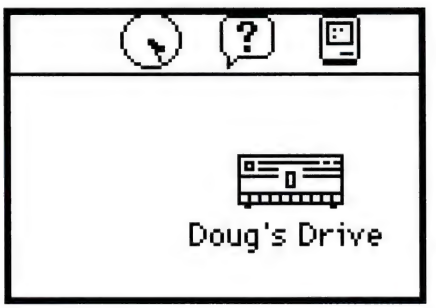


FIGURE 3.14 THE CLOCK IN THE MENU BAR

Scott Schmitz built Analog Clock as an alternative to SuperClock. Some people like the old way of doing things. Scott Schmitz dislikes digital clocks. He prefers instead an old-fashioned Analog clock, the big hand is on the...

Scribble

7 §

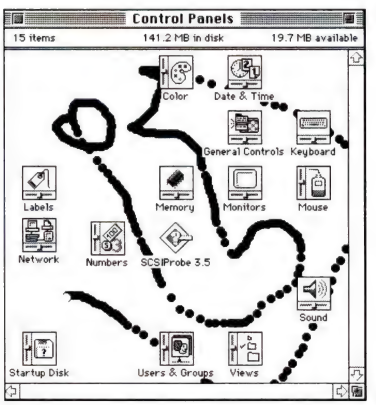


FIGURE 3.15 A "SCRIBBLE"D WINDOW

Not satisfied with all the changes that had made it into System 7, Darin Adler and Chris DeRossi had another trick up their sleeve. They added a pencil and an eraser to the Finder. The purpose? So that one can scribble away that the windows in the Finder. Make your own graffiti to leave notes and messages or just to confuse people about what you were thinking. The **Erase All** command returns the desktop to the same state it was in before the scribbling started. It also makes a handy memo pad when on the phone; now, if the mouse would just move like a pencil. Remember that scribble will only paint in the open windows, not on the desktop itself. It also will not allow a complete white background in the Finder. On an extended keyboard, use the **Command** key and the **Plus** key (from the numeric keypad) to cycle through tools and the **Command** key and **Minus** key to turn Scribble off.

SFComment 0.5

7

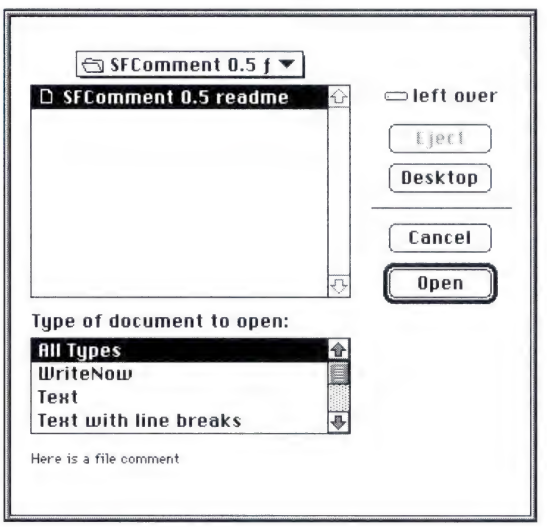


FIGURE 3.16 COMMENTS IN THE OPEN FILE DIALOG BOX COMPLEMENTS OF SFCOMMENT

Benjamin Waldman of Microsoft was disappointed that the file system under System 7 supported file comments, but that the Open and Close File dialog boxes did not support the comments. Comments are those little statements that are placed in the get info box on the desktop. Ben put together SFComment so that the file dialogs would show the comments attached to files and volumes. With the number of files in the system constantly rising, SFComment is more than handy. Now if the comments would just stay put when the desktop is rebuilt.

StandardGetFiles

7

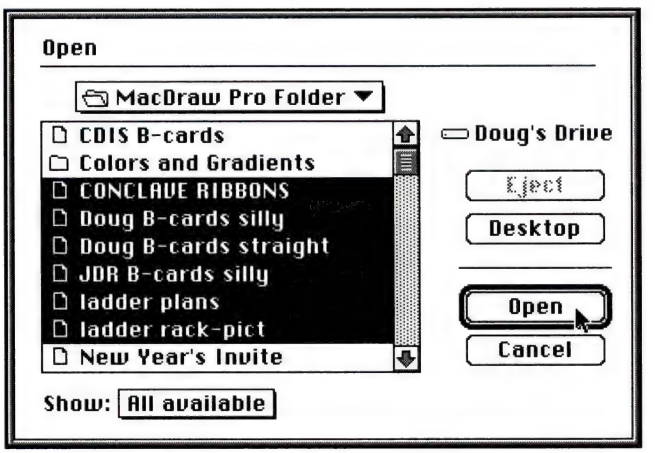


FIGURE 3.17 SEVERAL FILES SELECTED USING STANDARDGETFILES

StandardGetFiles by Dean Yu solves an old problem of opening a number of documents at the same time. If all the documents are in the same folder, StandardGetFile will allow the user to select all the files of interest at once and open them. The results of holding down the **Shift** key and clicking on a set of files is shown in Figure 3.17. The only enhancement that this hack could use is the ability to select files that are not contiguous in the file list.

Task-It

7 6 040

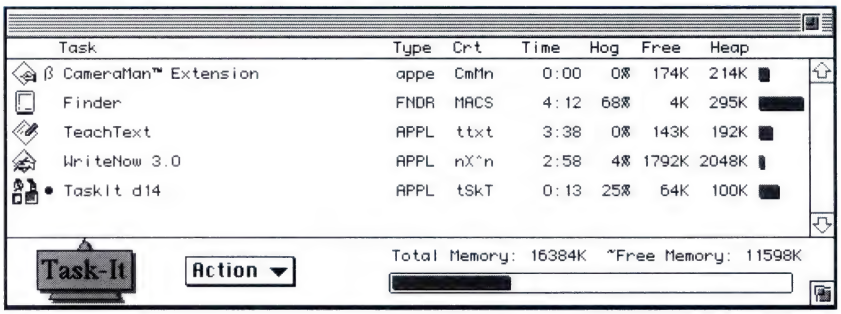


FIGURE 3.18 THE TASK IT WINDOW, COMPLETE WITH EXTENSIONS

Task-It by Pete Helme of DTS is an application that takes advantage of the process manager. It monitors CPU and memory usage and uses this information to create a window (see Figure 3.18) that includes information on what the machine is doing. It can see extensions and Control Panels. It will allow the user to kill a process, even if it is an extension that is normally not killable. Very useful in determining whether or not the memory level for a program is set correctly. If the program is in full use and it has free memory, the memory level is set too high. If the application does not have free memory, then there is a requirement to increase the memory a little at a time until free memory shows.


WARNING

Be aware that killing some processes can and will crash the machine. Many applications use two or more processes, like Photoshop scanning in a photograph. Killing any of them can crash the Mac.

The Grouch

7 6 ~~040~~

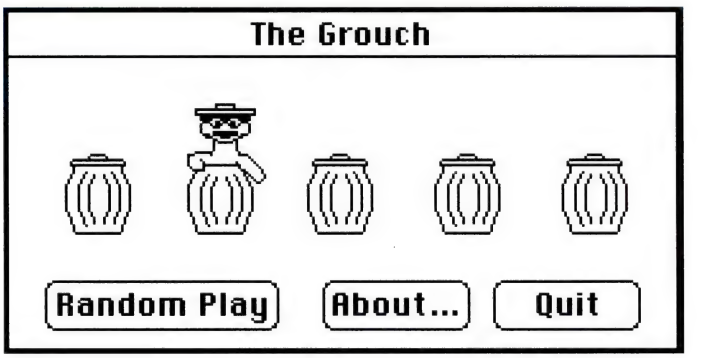


FIGURE 3.19 THE GROUCH NOW HAS FIVE HOMES TO POP OUT OF.

Eric Shapiro entered the grouch for the third straight year. After a large number of electronic mail requests and letters from parents, Eric built a standalone version of the Grouch for children, so that they would not throw everything on the hard disk in the trash. Notice that there are five trash cans to choose from in Figure 3.19.

Too Many Lawyers...

7.0.0 ~~040~~

Apple legal pushed to remove the Apple logo from third-party products during early 1991. David Koziol had had a run in with one of the lawyers. His solution to the matter was to create Apple Smasher AKA Too Many Lawyers. This extension removes the Apple Symbol from documents. If it is loaded and someone types the option-shift-k key combination in the Chicago Font (The Apple Logo) Apple Smasher goes into action. Lots of silly fun.

TrashMan

7

TrashMan by Bill Johnson and Ron Duritsch is a smart sanitation engineer for the Macintosh. When a user holds down the **Option** key and drags files to the trash, TrashMan automatically deletes them from the trash. TrashMan in later versions allows the user to select the minimum amount of free space on the disk, which would trigger the automatic empting of trash. TrashMan has been updated several times, and the current version is available on the various electronic networks.

VideoBeep by Eric Shapiro

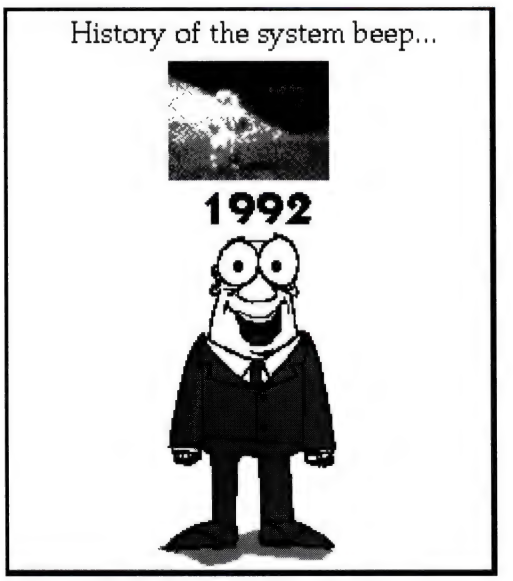


FIGURE 3.20 THE VIDEO BEEP ANIMATED **ABOUT** BOX, COMPLETE WITH A MOON WALK

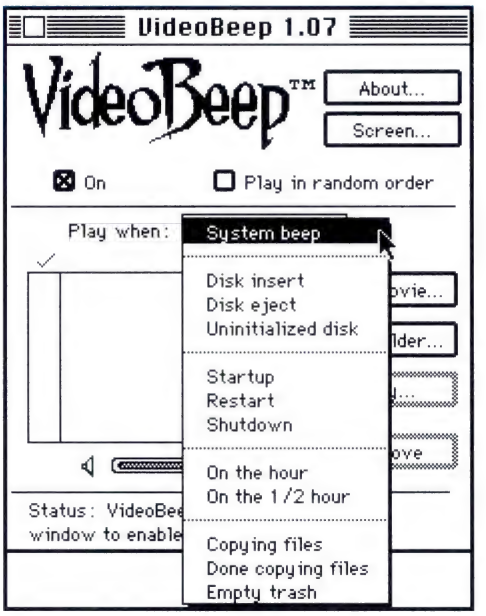
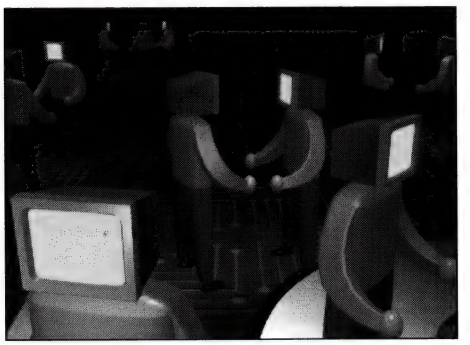


FIGURE 3.21 THERE ARE MANY CHOICES OF WHERE TO PLACE VIDEO BEEPS

Video Beep is shipping as a commercial product from Sound Source Unlimited with clips from *Star Wars* and *Star Trek the Next Generation*. This hack replaces the standard Macintosh audio beep with a QuickTime clip. Video Beep can be fun, but it also takes time to run whenever you get a system beep or another event that Video Beep can be configured for (see Figure 3.21). The Video Beep **About** box was a bit of fun at the hack contest because the sampled sound levels were so low in Video Beep, everyone was yelling at everyone to be quiet. The About box had to be played several times before everyone had a chance to hear it.

CHAPTER 4



1992—Tornado Hack

MacHack 1992 really got off to a fast start. Several major items were settled in the month immediately following MacHack 1991. In discussions with Sheila Brady and Kirk Loevner after MacHack, Roger Heinen the VP of Apples Software group was suggested to the committee as keynote for 1992. Roger was contacted; he accepted and was advertised as the keynote for MacHack.

Several MacHack committee members wanted to experiment with and try a new method for scheduling sessions, they wanted to run sessions around the clock. Also, there was a feeling that too many sessions had been scheduled at the 1991 MacHack. This led to a schedule that did not allow time for hacking. For one group of attendees, hacking time was the major reason for coming to MacHack. For others it was a chance to have hands on new technology and to get directly to the people developing the new hardware and software. In 1991, the content faction had won the right to schedule the conference and had filled the schedule. The sessions all started at the same time and offered many competing items. The major portion of the schedule was completed during normal business hours, but many of the programmers groused about the fact that there was content before lunch. The

other problem was that there were long lines at lunch because all the sessions ended at the same set time. Scott Boyd, a member of the Blue Meanies, was appointed as the Program Chair. Scott was a member of the *Hacking is the reason for MacHack* faction and offered to provide a radically different schedule that would have fewer sessions, many of which would have only one speaker. This schedule would usually have only one on-going session. Since Scott had his hand on the pulse at Apple, the sessions were in tune with Apple's goals with the Macintosh.

Allen Foster as the Conference Chair and Waldemar Horwat agreed to round up the papers again. The 1991 program had as many as five topics competing for attendees' attention. Chris Allen was a sysop on America On-line (AOL) and formed the MacHack forum on AOL. Chris convinced AOL to become a sponsor of MacHack.

During the year, MacHack was able to rebuild the depleted ranks of sponsors. Apple Computer became a sponsor and provided many items, including the keynote and development software that were required to make the conference happen. APDA provided goodies for the banquet and give-away prizes at the door. The University of Michigan provided the machines for the machine room. Jersey Scientific offered prizes for the Hack contest, which was a first. MemoryBank and John VanRoekel, one of the founders of MacHack, offered use of a large disk array for the server. MetaTec offered use of their facilities to make a CD-ROM with the proceedings of the conference for distribution to the members. This increase in sponsorship took pressure off the need to raise the conference fee or to increase attendance. MacHack could stay MacHack for another year.

The conference had a committee that was in high gear from day one. Many people were working behind the scenes. At Apple Scott Boyd, Allan Foster and Jordan Mattson were coordinating most of the sessions. At MIT, Waldemar was busy rounding up more papers than ever before. Ellen Zalman was busy building a set of programming items dealing with the increasing complex legal environment that developers had to cope with. Ellen, a lawyer and a hacker, wanted to make the legal aspects of developing software clear to the attendees. Brad Serbus was working with Carl Berger at the University of Michigan to complete the best machine room ever. Paula Smith and Gerry Felipe

were rounding up local volunteers to watch the machine room and do the grunt work that was required to make the conference successful.

People were very interested in MacHack and read about it in articles by Steven Howard and Raines Cohen in *MacWeek*. Steven Bobker and Ben Templin from *MacUser* discussed MacHack and its impact. Gregory Merriman wrote a piece for *Byte*. Of the publications devoted to Macintosh, only *MacTutor* and *MacWorld* were not represented at MacHack. Brita Meng who was the Technical Editor at *MacWorld* left the magazine for Shiva, and *MacWorld* did not have a replacement. Even *Nautilus*, the magazine for the Macintosh on CD-ROM carried news about MacHack. AOL and Internet posted the hacks.

Apple asked the committee to do a European version of MacHack as well as have regional MacHacks. Apple heard positive responses from the MacHack attendees and saw results in the software that the MacHack attendees were creating that other Macintosh programmers were not obtaining. Apple wanted to spread this kind of small conference around the globe and excite and empower other programmers. Weeks were spent discussing the opportunities; in each case, the answer to Apple was, we are sorry we cannot fill that request. The time and money requirements of taking MacHack on the road were just too much. There was a move to rotate MacHack around the country, but finding sponsors like the University of Michigan that can offer more than 100 state-of-the-art machines for use in the machine room was very difficult.

A scheduling change was made to move the Bash Apple session to the end of the conference and rename it the Apple Feedback session. This and other scheduling changes gave MacHack a different flavor. Programmers ended up spending far more time behind the machines and less in discussion. Teamwork efforts decreased and the exchange of useful information declined.

As WWDC approached, Roger Heinen backed out of his commitment to MacHack. He offered instead Steve Weyl, who was the Director of Developer Tools as a keynote speaker. The committee had little choice but to accept the offer of Mr. Weyl. Scott and Jordan assured the rest of the committee that Steve would be an excellent speaker.

The rumors before WWDC indicated that Apple had a large amount of new technology to talk about and that this technology in many cases was ready for trial usage. People went to WWDC with high expectations. There were going to be a number of stunning announcements. There were rumors of a major upgrade to System 7: A new architecture for the MPW programming environment, QuickDrawGX would finally be released, Apple had talking computers and palmtop computers, the PowerBooks were going to take a major step forward, and the operating system was finally going to get a protected heap and threading. All this was major news, and if it all happened, the Macintosh operating system would not only eclipse all other commercial operating systems, it would also be capable of outperforming UNIX in many networking and multitasking functions.

Apple discussed in detail the now famous PowerBooks and where they were going with them. They also described for the first time the enabler technology that adapts the system software to new machines. The developers did not like the idea then and many still do not. The enablers offered an easy way for Apple to customize the system, but because they are numbered and not named, without a list, there is no easy way to determine which enabler is required for a machine. Also, because the enablers are numbered, figuring out which version of an enabler is installed is a little tougher. Finally, the enablers were bad news because the people who needed to ship boot disks had to figure out how to get all the enablers on the disk. There were discussions of AMBER, now renamed OpenDoc. Bento a compound document architecture was also shown for the first time. Both of these technologies offered developers ways to work together and to make their applications work together. The year 1992 marked the first real announcement of the PowerPC and what the whole Apple/IBM/Motorola alliance was about. This alliance promised to be the biggest change at Apple since the introduction of the Macintosh in 1984. After a decade of small improvements, Apple was again going to make a quantum leap in the technology. Taligent and Kaleida Labs were going to provide the building blocks for the future multimedia operating system.

Apple's advanced technology group introduced Casper at the annual ATG session. This technology is the heart of Apple's PlainTalk software. The programmer was able to talk to his computer, and the

computer would respond by executing the programmers' instructions. Casper became the basis of the AV technologies. Beyond that, the attendees were treated to the fourth annual what's new in this year's version of QuickDrawGX, which, by the way, is not shipping this year. An update on font technology and other system software information were also described. Although this conference was billed as representing new, available technology, not a single item was offered to attendees. Most of the veterans grumbled, and it even annoyed many of the new attendees. This was the worst WWDC since 1986. The 10 goodie tickets that were attached to the badge were used only to pick up a t-shirt and to get into the party at the end of the conference. Many of the old hands spent much of the conference out in the lobby discussing the problems with Apple. Roger Heinen told the programmers to bet their companies on Bedrock and OpenDoc. That they would be the most important programming tools and advances in 1993. The veteran programmers laughed at this statement, which became the tag line for the whole event, bet your company on Apples vapor.

There was a feedback session about WWDC at the end of Friday. The attendees pointed out that offering the equivalent of propaganda to attendees that were paying over a thousand dollars each, was not the way to ensure cooperation with the developer community. The session dealt with the increasing costs to be a developer and the reduced amount of information that that status brought to the developers. It was also pointed out that without code to take home, there was no sense in betting the company on technology that Apple was going to offer, because no one could use what they did not have. Roger Heinen grew uneasy while Steve Weyl took careful notes. Steve answered most of the questions. The session had an angry underlying tone that carried over to the party that night and even to MacHack.

There were five weeks before MacHack; Scott had played the MacHack schedule close to the vest, telling only the Apple employees who were speaking. With five weeks to go, the rest of the committee had its first look at the schedule. It was very heavy with Apple speakers and did not offer the mix of Apple and industry that had happened in earlier years. The schedule did have more leading edge and new technology than ever before. The sessions were scheduled carefully, focusing on the most important items. Scott had built a very different schedule

with great care, and it looked like it would work very well. The speakers were allowed weeks to prepare, which led to an incredible number of slide presentations and formal presentations on topics of interest. In several ways, having a light program was better in 1992; there were no earth-shaking topics that needed to be covered outside of AOCE and OpenDoc.

Many of the MacHack committee members had their first chance to see Steve Weyl in action at WWDC, and all agreed that he was a great choice for the keynote, better even than Roger Heinen. Roger's area assistant listened to all the MacHack committee members' complaints at the closing party for WWDC and promised to put all of them on beta test lists for all the material that was promised but not delivered. The party was noisy, and the promise was made very late in the evening. It was not surprising that nothing ever came from this promise. It and many other WWDC promises never materialized. Any developer who followed the 1992 garden path is likely to be broke and out of business today.

The committee was so well organized that there was no need to scramble during the time from WWDC until MacHack. The conference was filling up, having benefited from the publicity during the fall and winter. The speakers and equipment were all lined up and ready to go.

MacHack '92

Brad Serbus was the first person on-site Tuesday morning. His trucks appeared early; he and his crew were not going to endure the same problems that they experienced the prior year. Brad and the University of Michigan crew had the machine room set up and ready to run in record time.

The Hack contest was overwhelming. By Wednesday morning, before the official start of the conference, a number of hacks were already in progress. There was one prediction that over 100 hacks would be entered. If the prediction held, there would be no sleep until after the contest ended Friday afternoon.

MacHack began with a session called MacHack 101, a light-hearted and funny introduction to what happens at MacHack and how to make the most of the conference. There was talk that it ought to be taped for the Comedy channel! Brita Meng and Allan Foster both have a great sense of humor and play off each other very well. When you throw in a room full of programmers who are all capable of providing alternate punch lines and puns, the session can get very silly very quickly. This session put the attendees in a great mood for lunch and the keynote that followed.

In his keynote address, Steve Weyl did the things Kurt did not do. Steve understood his audience, did not try to wow them with marketing numbers, and instead focused on issues about the Macintosh direction, system software futures, and the importance of great tools. He stayed the entire week to work through the sessions. Whenever Steve promised an answer, he provided it. He was on the phone to Cupertino several times and sent several batches of e-mail. In many cases, the promised answer came before the end of the conference. In others, his answers were delayed for lack of information. In every case, if he promised an answer, he provided it. Steve also provided the most important quote at the 1992 MacHack: he told the audience that good programmers have a tool that most people do not have, that is, SmartFriends. Since then, it has turned into a line of popular t-shirts.

The question-and-answer session started out hostile, which was expected given the disappointment of WWDC. But the session ended on an upbeat note. Steve Weyl had done something that no one since Doug Clapp had done, he won the respect of the Macintosh programming community. Steve has since credited this session with the development of the Apple Directions CD that was issued at the 1993 WWDC. His commitment to MacHack and the programmers there had great impact on the way that WWDC was conducted in 1993.

Steve's session was followed by David Shayer's, the developer of DiskLock for Fifth Generation System. The session was entitled *Mating Rituals* and covered the ins and outs of finding a publisher and how the business side of getting a product out into the market was

handled. Eric Shapiro was the other panelist who offered insight into the negotiations with several of the publishers.

Five non-Apple programmers then tackled that whole problem of QuickDraw GX and how it was going to affect programming on the Macintosh. They looked at the long and tangled history of GX and offered hope that GX might see the light of day in 1994, not late 1993. The problems with writing a program for GX and also keeping the program compatible with the old version of QuickDraw were discussed in detail. It was obvious to most of the people in the room that GX would require major rewrites of the programs that were already shipping to take advantage of GX. With the changes in the way the QuickDraw commands were called and the different structures for GXs calls, to take advantage of the new features meant major rewrites of many popular applications. What was not clear to the panel was whether GX was worth the effort of rewriting their software. Steve Weyl sat quietly in the back of the room taking notes, which led to several long answers to questions about GX in *Develop* magazine and tutorial material on the developer CDs. This also led to the development of a set of classes for C++ to make using GX easier for programmers.

Later that afternoon, Scott Boyd and Greg Marriott offered insight into the Hack contest, how to enter the hack, what kinds of things won, what the prizes were, and how the whole contest was put together. They showed past winners and discussed why those hacks had won. Again, the session was presented in an upbeat and light-hearted manner.

Waldemar Horwat took both the Macintosh Operating System and Microsoft Windows to task in the late afternoon. In his typical understated style, Waldemar proceeded to tear apart each of the operating systems and look at the advantages and disadvantages of each from a technology and programming standpoint. No one in the room claims to have understood everything that Waldemar said, but his paper again spoke to the technical points (one point he made was on structure). Waldemar considers the question of object-orientation in the following excerpt from his paper:

Aside from the language syntax (which is important), one can also ask whether the structure of the operating system is object-oriented. By this I mean whether system structures are presented as abstract, self-contained entities that can only be manipulated by functions and from which one can derive subclasses.

In this respect, Windows has a more object-oriented structure than the Macintosh operating system. Windows communicates with application windows and controls by registering an applications entry point and then sending it messages as appropriate. If the window or control does not wish to do anything special with a message, it can pass the message to a default handler specified by Windows. Thus, each window and control is effectively overriding a default. Windows also allows defining controls by overriding the behaviors of predefined controls, but this is difficult and has to be done by trial-and-error for the reasons stated in Appendix A: When changing one area of a window by overriding the draw message, one cannot be sure that the window doesn't make assumptions about the size or shape of the area being changed. Fortunately, the default window and control functions don't seem to be doing that, but they could without violating any rules in the documentation.

The Macintosh is more procedural, in that the operating system relies on the program to dispatch events and has no direct access to a program's windows and controls. This is a disadvantage; for example, when a Modal dialog box is shown, the program's event loop is not executing until the dialog box is dismissed, so that none of the programs other windows can be updated. If part of another window belonging to the program becomes invalidated, a partial deadlock will occur. The work—around is quite messy .

Windows object design is better for most applications, it is not superior in all cases. One trouble spot in Windows implementation of messages is quitting a program (or, worse, quitting Windows). The program gets one quit message and is then expected to instantly decide whether it wants to quit, and, if so, do it cleanly without receiving any

more messages. This is fine if the main program's event dispatcher gets the quit message it can just exit the program. On the other hand, if a dialog event loop gets the program's quit message, then it is difficult to cleanly unwind all of the pending calls on the stack precisely because the program is not procedural; the dialog's only choice is to return to the Windows dialog handler, but when that handler returns to the place in the program where the dialog was invoked, the program wont know that it is supposed to quit.

This is one of the best pieces that has ever been attempted on this issue. Many authors since have quoted from this section and others in the paper.

Bad Weather

Wednesday night was the annual banquet. This year it was a barbecue out in the open, the year before the banquet had been cut short by a rainstorm. This year, when the buses arrived, the sun was out and the day was warm and sunny. It looked like the perfect evening for a picnic. The chicken and tofu were well done and everyone was having a great time. Henry Norr, of *MacWeek*, was circulating, asking questions about the conference and attendee's real jobs. Steve Weyl was moving from table to table taking questions and offering answers where he could. The sun was shining, but the breeze was beginning to get cold. The sky was rapidly filling with clouds, the sun seemed to just disappear. First, it just sprinkled and the discussions continued unabated. Then raindrops spit from the sky with increasing frequency. Everyone boarded the buses, just as the sky turned bright grass green. In less than 15 minutes, dry roads had puddles that were inches deep. The clouds turned day into night. The thunder rocked the buses and the lightning flashed turning night into day. The driver on one of the buses announced over the radio to the others that there was a tornado in the area. As the buses pulled into the hotel, the staff was directing people to the interior hallways as a safe place to be. As might be imagined, the machine room was closed for the duration of the storm. Several times during the course of the evening the power flickered on and off.

Two tornadoes touched down in the area that night, one a couple of miles from where the banquet had been and one a mile from the hotel. In both cases, no one was killed. The machine room was quiet for most of the night for the first time in MacHack history.

For part of the evening, the Stump the Experts panel was held in the hallway, the balance was held in the room where it had been scheduled. The best part of the evening was that a number of programmers developed new friends and teams were formed to create several of the hacks that were presented Thursday night.

The morning broke with heavy clouds and intermittent thunderstorms. Several times during the day, the staff came through the machine room to shut it down until the thunder and lightning moved away. Many of the West Coast attendees had never experienced a thunderstorm of this intensity, hence they were nervous and spent much of the day in the shelter of the central hallway. At times, it was hard to travel the length of the hallway without stepping over several people. Eric Shapiro captured many of the more interesting moments from the storm on file and moved them to QuickTime movies. (Those movies are on the CD).

Thursday's panels started with an AOCE panel. Steve Falkenburg and Keith Stattenfield of DTS are two people at Apple responsible for this technology. They went into detail about what AOCE was going to be about and how to make use of AOCE. There was discussion of AppleScript and AppleEvents and how both of these technologies were building blocks to the AOCE world. Steve and Keith had a lively debate with a number of developers about the uses for AOCE and how most developers felt that the big companies would not take advantage of it, so the little developers could not really take advantage of it. Without the support of the large developers, the small developers could not create the niche products that would work cooperatively with the major productivity packages. The session ran well over time and was moved to the lobby bar for wrap up.

Ben Waldman from Microsoft spent an hour with a very skeptical audience on how OLE was going to be a major technology for cross-platform applications and how it would impact the Macintosh. Microsoft had been the only major Mac software company that would

not run with System 7. Microsoft had always followed a different drummer, and the prevailing thought was that Microsoft was again trying to put one over everyone. Everyone acknowledged Ben's understanding of the technology, and he spent most of his time on coding and coding details, this preventing him from getting the same treatment that had befallen Kirk Loevner the year before. Ben ignored the side comments and cat calls and stuck to the technology. He acknowledged that this was a Windows technology first. By the time the session was over, everyone acknowledged that Ben was a good hacker, he was just misguided in his choice of employer.

Sean Parent of Apple spent a large portion of the afternoon filling people in on the new PowerPC. He gave an overview of the new microkernel and how it would be used as the basis of the Macintosh software for the PowerPC. Sean went on to talk about the emulator for the PowerPC that would allow stock Macintosh software to run on the PowerPC. This meant that software written for the traditional 680x0 Macintoshes would run without a problem on the RISC-based PowerPC Macintoshes.

Eric Shapiro tackled QuickTime later in the afternoon. He explained how to use the software and showed the improved versions of VideoBeep and Spectator, his two QuickTime hacks that had become commercial products. He was followed by Rick Fleischman from Apple who discussed the future of MPW. The talk Rick gave was upbeat and discussed a dynamic future with vast improvements in MPW and MacApp. There was discussion of versions of C++ and C for the PowerPC, which would be complete before the end of 1992 or at the latest by the developers' conference in 1993; ready in plenty of time for the PowerPC roll out. In retrospect, Rick had been lead down a garden path, and he in turn provided that same garden path, to the attendees.

Midnight rolled around, and there were a number of jokes about the tornadoes and the thunderstorms. The Hack contest was off and running. There were silly hacks like the Annoyance Pack and serious hacks like the ARAStatus hack, with each hack presented to the audience one at a time. The machines in many cases had to be rebooted over and over again as each hack was loaded, shown, removed, and then the cycle was started again. Since the hacks were not tested with one another, loading two at a time was risky. Each cycle was displayed on

a large movie screen where it was projected by a large video projector. The room was dark and warm. As usual for a hack contest, a number of the hacks crashed. This of course brought cat calls and shouts of encouragement. The hack contest was described by Leonard Rosenthal late the next morning as NetBunny on drugs, it just kept going and going and going. There were over 70 hacks shown and over 90 entered in the official hack entry stack. Someone asked about quitting when the sun came up. The show lasted 4 hours, a new record in Hack contests. Greg and Scott had a good time hosting it and the audience had a great time watching. On this occasion, the weather did not interrupt the evening festivities.

Eric Shapiro—Teacher and QuickTime Hacker

A native of Michigan, Eric Shapiro was a student at the University of Michigan when MacHack was founded. Eric was interested in the Macintosh and attending the local user group, MacTechnics, meetings when an announcement that they were looking for volunteers was made in 1987. Eric volunteered at that MacTechnics meeting to assist with MacHack. He helped out in a number of ways that year, from setting up the machine room to helping on one hack.

By 1988, Eric founded Rock Ridge Enterprises, named for the town in Blazing Saddles, and was a consultant to Irwin's Magnetics, where he was one of the key programmers behind Irwin's EasyTape for Macintosh software. EasyTape was one of the first backup products that allowed the user different views of the material on the disk. It won several awards and was a year or so ahead of its time. Eric might still be working on EasyTape, if Irwin had not been bought by Cipher Data Products. Eric moved on to work on a series of products from medical software to databases.

Eric also used MacHack to hook up with Apple's Developer University. Eric taught for Developer University for several years, developing courses and modifying the ones he did not develop. Eric was in great demand by companies to come in and teach Macintosh programming. Like Richard Clark, Eric has a great sense of humor and

understands the needs of his students. Eric coaches soccer and softball teams, in addition to being a regular speaker at the MacTechnics meetings. He is always ready to help someone with a problem, with no financial expectations.

Eric's first hack for MacHack was Oscar, later renamed the Grouch. This hack is typical of his view of life. Eric likes to do things that are fun and that other people can enjoy. Oscar made three trips to the Hack contest, and even on the third trip in 1991, people still enjoyed seeing Oscar. Eric did not stop with Oscar. In 1990, he added Eric's Menu Hack and Eric's ColorWheel to his list. Again, he showed in an enjoyable manner a deep knowledge of how the system software worked. In each of his hacks, he developed an elegant method of changing the function of the operating system. ColorWheel allowed the user to place the wheel on the correct screen each time. It meant patching the operating system to remember the location of the wheel. It also meant keeping that preference around after the machine had been shut down and then restarted. The menu hack required a complete understanding of how the operating system built the Apple menu so that he could modify the way it operated. Hooking up the folders in System 7 and building the structures so that they worked correctly was not a trivial matter.

In 1991, Eric conquered QuickTime faster than almost anyone else and built VideoBeep. This was the first of Eric's hacks to become a commercial product. The second hack to turn commercial was Spectator, a screen recording package, that required a lot of intelligence on how the screen operated and how the video was stored. In each case, Eric proved that with very little background he could use new features of the operating system to build programs.

In 1992, Eric built an Adobe Premiere filter that could be used to fuzz out a face or another moving object, much the way the real-life police show fuzzout faces to protect the identity of the person that is being arrested. In 1993, Eric did not enter a hack, he was too busy getting products to market. He is, however, working on his 1994 and beyond hacks.

Eric has been involved in MacHack in many ways. He has served on the committee, he has provided names of people to contact about

speaking at MacHack, he provided the first set of MacHack fortune cookies, he started the midnight movie on Friday night expedition, and finally, he did all the coordination with MacTechnics to make it possible for MacTechnics to become involved the first time and to remain involved with MacHack. Without Eric, MacHack would not be possible or nearly as much fun.

The 1992 Hacks

3D Bouncing Ball AD Module

67 §

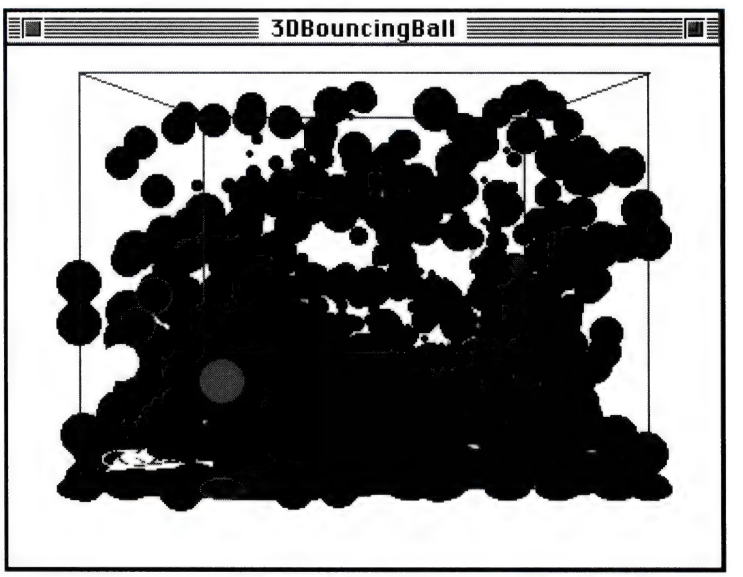


FIGURE 4.1 THE BALLS IN ACTION WITHOUT GRAVITY

3D effects on the computer screen can cause people to wonder if the computer is real or if it is a fishbowl instead. With 3D Bouncing

Ball, up to five balls bounce around in a room (see Figure 4.1). The balls can be effected by gravity and have shadows, or they can ignore earth physics and just bounce around. These rendered balls are lots of fun to drop on an unsuspecting machine.

Annoyance Pack

67 §

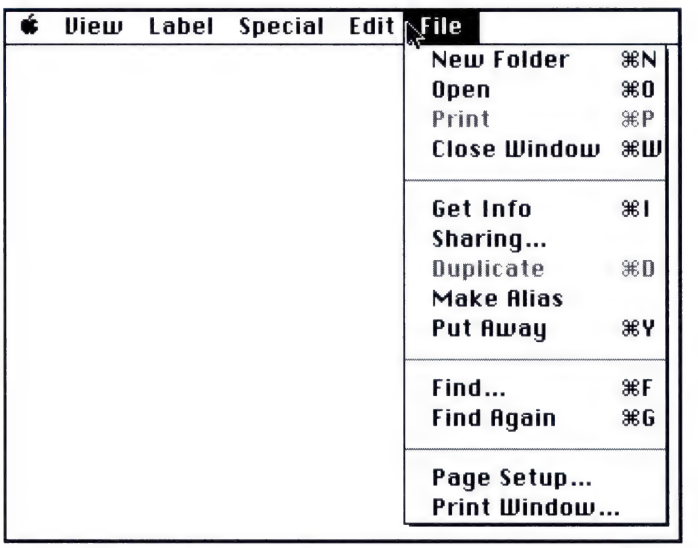


FIGURE 4.2 MENUHACK SCRAMBLING MENUS

Bad day at the office? Boss have you down? Want to get even? Here it is, the Annoyance Pack, the winner of the silliest hack award for 1992. Shane Looker wrote five extensions that annoy users and will turn a Macintosh into a 2 year old.

- *DOS-Not!* will not allow the user to type the word DOS.
- *Hot Shift* moves the cursor's click point or hot spot about half an inch up and to the left, which can be terribly annoy-

ing when trying to click on a document icon because the document will only occasionally launch. Most icons are large enough that the user clicking in the lower-right-hand corner will open the document and when clicking anywhere else will not.

Eggs and Menus both can be scrambled, and Shane offers three different ways to do so. Methods from *it is just not there (no redraw)* to the *menus move (rotating)* to the *straight menuhack* (see Figure 4.2). This is certain to frustrate people who are on a tight deadline. Menus will not continue to move, if the mouse button is held down while moving in the menu bar. There is no icon displayed when loading this extension.

NVwl (No Vowels) is a hack that removes all the vowels from anything that is being typed at the time (try t sm tm).

Squeaker II makes the Macintosh squeak loudly every time the mouse moves or the mouse button is clicked.

AppleShareSetup

6 7 SW-AppleShare 3.0

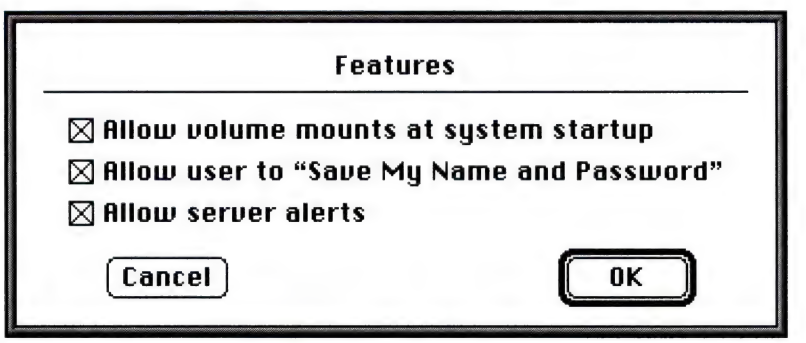


FIGURE 4.3 NOTIFICATION FEATURES

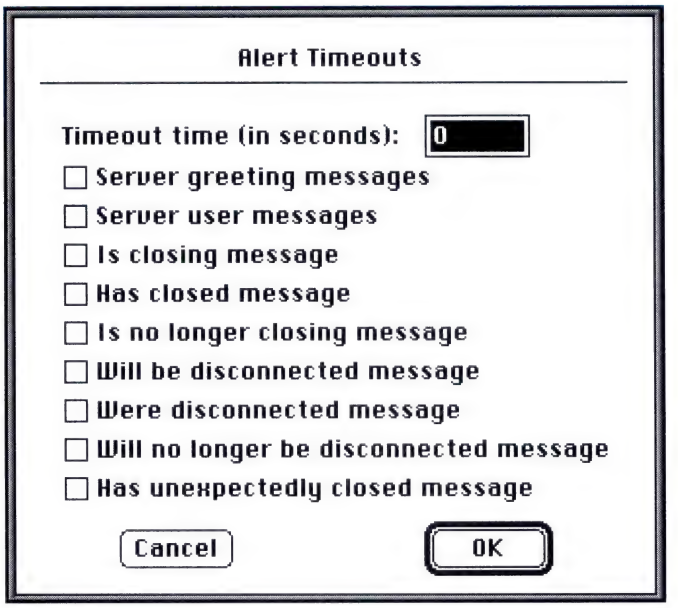


FIGURE 4.4 TIMEOUT AND WARNING FEATURES

Jim Luther is the DTS technician assigned to support networking. With AppleShare 3.0 Apple had a large number of features that were not documented. Jim built this Control Panel to access those features. Figures 4.3 and 4.4 show some of the options. If a mistake is made in setting up the server, all of the settings can be returned to the factory settings with the Default button.

AppletalkOff II

7

When a PowerBook is regularly moved about, it has a tendency to use the network. AppletalkOff senses whether or not the PowerBook is attached to a network and turns AppleTalk on and off automatically. Keith Stattenfield, who wrote AppleTalkOff, works for Apple's DTS group.

ARAStatus

6 7 § SW-ARA

ARAStatus by Patrick C. Beard and Chris Allen is a useful hack that lets the user know whether or not the *AppleTalk Remote Access* (ARA) connection is working and if so, for how long. ARAStatus quickly tells the user whether the network connection over the modem is solid and ready to use. It helps troubleshoot lost files and very slow connection rates. This one was written in frustration as the ARA connections from the hotel were always going down.

Battery Indicator 140/170

7

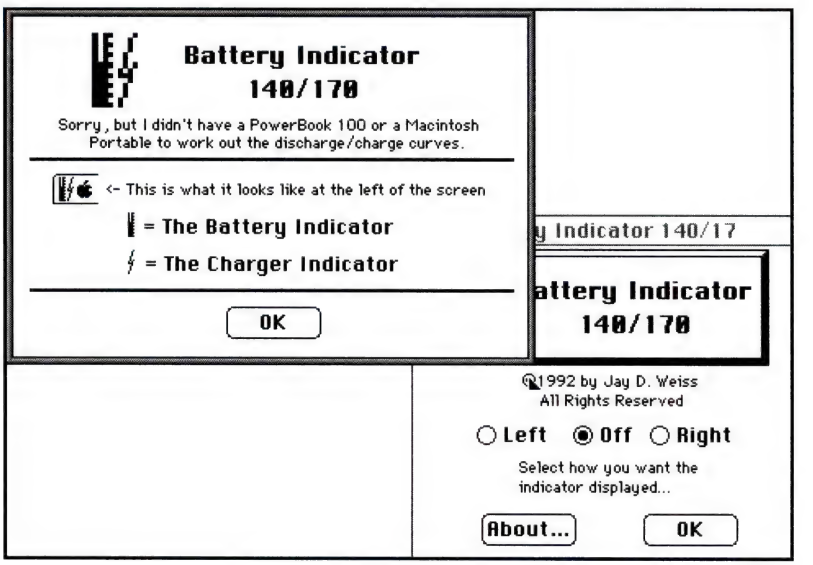


FIGURE 4.5 THE BATTERY INDICATOR CONTROLS

Battery from Apple is one of the easiest parts of the PowerBook to bash. Only the plastic door over the ports seems to take more abuse from users. Jay Weiss disliked Battery enough to write one of two alternatives (see Figure 4.5). He utilized the conventional extension approach, where the extension offers two location choices for the monitor icon, which can also be turned off. This version of Battery Indicator is tuned for the PowerBooks 140 and 170 and can be used, with less accurate results, on other PowerBooks.

BattMonitor

7 §

BattMonitor is the less conventional solution to replacing Battery on the PowerBooks. BattMonitor is an application that will only run on PowerBooks and Duos. The programmer, Christian Russ, did a great job of error checking, so that if the application is launched on a desktop Macintosh, the Mac will not crash, it will just seem like it for a few seconds. Click on the Continue button rather than the Restart button to avoid problems. BattMonitor checks the battery status on a PowerBook and provides a graph of how much time is left. It takes into account use of the hard disk, screen brightness, and so on. Unlike Jay Weiss's approach, BattMonitor will run on most of the current PowerBook models, giving accurate results. It is an application and should be placed either in the Startup folder in the system folder or the Apple menu folder.

Bell Choir

7 §

With a network of Macintoshes to play with, Katheen Brade took the idea of play literally. This hack was a strong runner-up for the best hack award. She created a pair of programs that allow the Macintoshes

on the network to play music much the same way a bell choir does. Each Mac is assigned a note, and it plays that note according to the instructions of the lead Macintosh. The instructions for loading the software are straightforward. Bell Choir requires Macintosh IIci or better to be run.

BlueDot Hack

7 § SW-Adobe Premiere

William Kennedy Smith was on trial during MacHack 1992, and Eric Shapiro created a video hack based on the trial. The news fuzzed out the face of the victim during the trial. Eric created a filter for Adobe Premiere that has the same effect. It can fuzz an area selected by the software user. It has some very silly applications in addition to the useful *hiding identity.*

BNDL Hacks

7 §

The *bundle bit* often changes on an application. Most of the launch problems (application cannot be found) and icon problems (icon changes from the proper one to a generic one) happen because of this. Mike Engbar tackled the problem with Save a BNDL and Flip A BNDL. BNDL is the resource type of the bundle bit. To fix the bundle bit conventionally, one rebuilds the desktop. To use the unconventional method, run Norton or Public Utilities. The hacker's way is to use ResEdit or better yet these utilities. To change the bundle bit, use Flip A BNDL; to reset it in the Finder, use Save a BNDL. Mike ended up working for Apple's Developer Technical Support group based on these and other hacks he created and displayed at MacHack. The bundle hacks are regularly updated on-line. These are drag-and-drop applications that take advantage of the drag-and-drop features in System 7.

Change Type & Creator

7 §

Change Type and Creator is one of the most useful hacks from MacHack. When downloading many text files, they normally will not open when double-clicked without some sort of dialog box about the application not being found or without offering TeachText or some other default. Change Type and Creator allows many files to be dropped on it. It then changes the file to make them appropriate for the program that it has been set to. To determine the correct type and creator, Brian Bechtel made the program recognize the current type and creator and then display them. So to use it, just drag a file that will double-click correctly onto CTC and then copy the information onto a piece of paper. Now drag the files that need to change and enter the type and creator information in CTC. CTC will take off and make the changes. Close and then reopen the file folder and the icons will change to the correct icons, double-click, and they open correctly.

Conan the Librarian

7 § HW-Sound Input Device

Quiet is the watch word in a library. Conan exhorts people to be quiet, when they are noisy around the computer. The louder the noise the microphone picks up, the more Conan works to get people to quiet down. This is a great party gag. A MacRecorder or another microphone and Conan are all that is required to set up this hack. Conan is an application, not an extension, so it is compatible with most machines. Conan can get fairly vocal when the noise level gets to be loud.

DylanTalk

67 §

DylanTalk started out as just a neat hack, but then it was shown to several visually impaired Macintosh users and it became a full blown utility. Reading dialogs can be very time consuming when vision is a problem. DylanTalk fixes this by reading the dialogs to the user and then the text of the buttons as they are clicked on. Dean Yu of NetBunny fame created DylanTalk with Fred Monroe. The extension has been overtaken to a large extent today by Apple's PlainTalk software. DylanTalk is still useful for older machines and for people who do not want the overhead of PlainTalk. The hack was designed to use silly voices, and it does to great effect. DylanTalk has not been updated and was written in a few hours; it is still very useful. The icons were changed to reflect the tornado that interrupted programming at MacHack by Dean and a few of his SmartFriends.

FinderMenu

67 §

Sometimes it is handy if a program menu appears in the Finder, that is, menus like the StuffIt Deluxe menus that offer significant additional functionality to the Finder. Steve Zellers offers a solution to this need. Finder menus and the associated source code that is with it offer users a chance to have a menu from one or more programs appear in the Finder. FinderMenu is built on the technology found in Frontier, specifically the Menu Sharing technology. FinderMenu was written mostly to avoid having to use the **Open** command to get to files from programs. To make full use of FinderMenu requires some programming, hacking actually to allow programs to offer menus that can be shared.

FlashBack

7 § HW-PowerBook

This hack will send flashing light messages over a network to other PowerBook users. It has two parts: FlashBack, which needs to be installed on every PowerBook in the startup folder in the system folder; and Flashem, which then controls flashing by communicating with FlashBack. David Shayer of DiskLock fame and Barry J. Semo, an Apple contractor, wrote it. Barry had previously written Strobe to flash the screens to save batteries. David and Barry wrote this one to win an award at the Hack contest. It can be useful for announcing a meeting, but that is about all.

HyperInitMaker

7 § SW-CompileIt!

Tom Pittman likes to do things that hard way. His choice for creating extensions is HyperCard. But can HyperCard be loaded at Startup? Tom created an extension that allows HyperCard stacks with XCMDs that are compiled with CompileIt! to run at startup. This was the winner of the sickest hack award (a major prize that is coveted by many programmers) in 1992. Making HyperCard stacks for use as extensions seemed to many of the programmers to be the long way around the problem. Tom has provided great documentation and the full source of a sample.

iconEdit

67 §

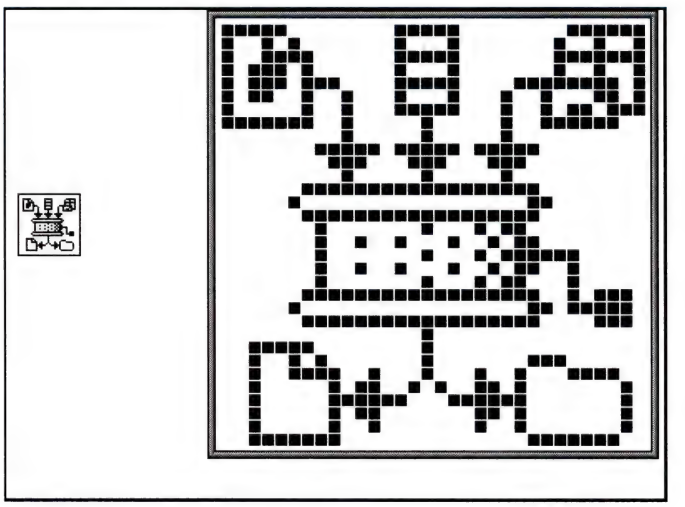


FIGURE 4.6 THE FATBITS SCREEN FROM ICONEDIT

Editing icons used to require a dedicated resource editor or ResEdit. With iconEdit, all that needs to happen to edit the Black and White icons is to open the Get Info window and select the icon. Once that is done, the fatbits picture of the icon will open and the cursor turns into a pencil. This hack was written by Steve Kiene, the author of Vise and several other Macintosh utilities.

Insomnia

7 § HW-PowerBook

Insomnia was written by Stephen Somogyi of *MacUser* magazine. Stephen normally writes the test applications for *MacUser* for some of the lab reports. Insomnia disables all of the idle and sleep setting on a PowerBook when it determines that the PowerBook is plugged into the wall. The extension has one bug, but the t-shirt offered in the read me document has already been claimed.

IR Man

7 §

Mike Neil's hack won him a job at Apple. He also won the Hack contest hands down. Using parts from Radio Shack and the directions that are included here, anyone with a little electronics skill can create their own IR Man. Mike's comments about IR Man were: "IR Man lets you control many of your Macintosh functions with a Sony remote control like the one that a VCR would use. This is a sick hack." There is a little hardware involved (1 chip, a resistor, and a 9-v battery). The hardware plugs into the sound input, and the waveform is decoded into the bit wise values. Currently, you can control an Apple CD-ROM, QuickTime Movies, and application layers. With this remote, Mike changed the tune playing on the CD-ROM drive from across the room, he opened and closed windows, and basically used the remote like a mouse.

LoonyHelp

7 § SW-MPW

Programmers can get very bored, which can be dangerous, or even silly. Looney help is silly. It animates the balloon Help icon in the menu bar, blowing the balloon up and then deflating it.

Momentum

7 §

Momentum 1.01 INIT was created by Jim Stegman, CDEV by Bill Johnson, and the idea came from David Griggs. Momentum is a physics experiment on a Macintosh. When a window is moved, it does not stop moving, instead it continues to move at the velocity that it was moved at. The CDEV allows setting the fiction and gravity information and the **Arrow** keys on the extended keyboard work like rocket engines, so that the window can be accelerated or slowed. Hitting any other key on the keyboard will stop the window cold. This is a great trick to pull on someone.

Mr. Bing

7 SW-QuickTime

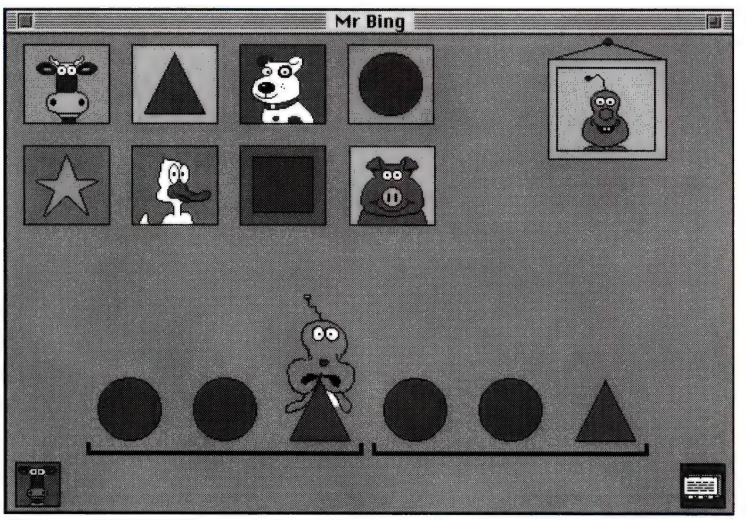


FIGURE 4.7 MR. BING, NOT THE AVERAGE FEATURE OF A CHILDREN'S GAME

Eric Slosser describes Mr. Bing as: "After months of working on a children's game which never scolds, never complains, and is always cheerful and supportive, I noticed a subtle change in my thought processes. The enclosed movie shows a hidden feature that I created to let off steam. I have promised the company for which I did the work that this particular feature would not appear in the release version."

NetMouse

7 § HW-network

When the author of Now Utilities sat down with the author of Intel's modem and networking software for the Macintosh, everyone wondered what they were doing. Eric Hayes of Intel and Jörg Brown of Now created one of the neatest hacks in 1992. If there is a PowerBook and a desktop Mac connected over a network and the two machines each have netmouse installed, the desktop Mac can control the PowerBook from the keyboard and mouse of the desktop machine; no more switching keyboards to move files from machine to machine. There is a good installation document to follow in the folder with NetMouse.

NetWarmer

6 7 § HW (see instructions)

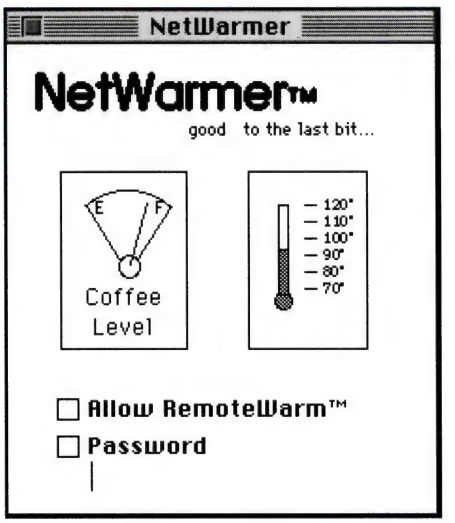


FIGURE 4.8 THE NETWARMER CONTROL PANEL, COMPLETE WITH A TEMPERATURE INDICATOR

Using the Control Panel and the installation information included with it, a simple network device can be constructed to warm coffee or tea on demand. This hack was written by two caffeine-starved programmers in the middle of the afternoon. Note that the Control Panel can be password protected to prevent device sharing over the network.

Network Digital Video

7 HW-Ethernet network, RasterOps STV card

Moving video and sound over the network can be time consuming, and the video never seems to arrive in real time, but rather, much slower. Mike Neal, the programming genius at the University of Michigan's Computer Aided Engineering Network (CAEN) provided a solution for the problem. This solution allows a RasterOps 24STV to send video over the network in real time to any machine that can receive it. The hack has two major parts: the DigVidSender and the DigVidReceiver. The sender is installed in the machine where the video will start from, and the receiver is installed in any machine that wants to see the video. This version of the software will only run in a zone named Holiday Inn Ann Arbor. Contact the University of Michigan for more information about the software.

Not

7 §

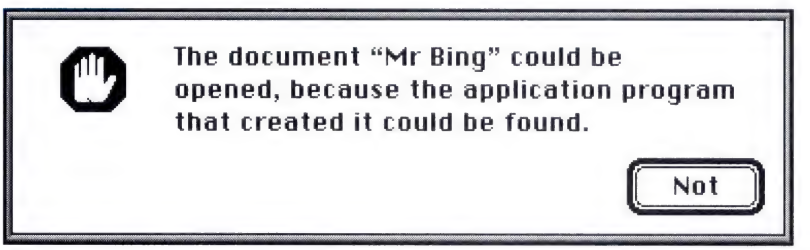


FIGURE 4.9 NOT IN ACTION, NOTICE THE OK BUTTON, NOT!

This is an all-star hack. This hack changes the OK button to Not and says Not! when the button is clicked. Several of the senior hackers had finished the hacks they were working on and started to discuss Wayne's World when the "Not" idea hit them. They sat down at a machine and began to program. Sitting around the Mac were Darin

Adler of General Magic, late of the Blue Meanies; Greg Marriott, the head Meanie; Jim Reekes, the father of Sound Manager 3.0; and Nevin Liber, Philip Nguyen, and Jeff Miller, all senior Apple software engineers. They were working on an idea that was originally brought up by Waldemar Horwat. The hack was completed (amazingly) in under 2 hours. It uses three patches to the operating system. Suggestions for improvements can be found in the Read Me file. This hack is limited to running under System 7.0.0.

not!

7

not! is another tribute (?) to Wayne's World. It is just resources and source code to add to your own projects. Eric Slosser figured that it would be the mode of speech for most of the MacHack attendees, so he prepared some code to assist in making the hacks speak the dialect also. The code replaces the Cancel button with a not! button and the Okay/Cancel button dialogs with a Way/No Way dialog. This makes your Macintosh Wayne's World savvy in addition to being System 7 savvy.

Open, Open, Open

7 §

When the machine is slow in opening a file and getting things going, there is a frustration level. Some of the programmers at MacHack can be heard chanting open, open, open as a file slowly makes its way from the overburdened server to the desktop. Eric Slosser decided to let them save their voices. He created the Open, Open, Open extension to let the machine do the talking. Eric minimized his use of resources, so that the machine would not slow down when opening the file. Save your voice and let Open, Open, Open do your chanting for you.

PB Keyboard Remapping

7 HW-PowerBook

Once your fingers are used to keys being in a certain location, switching keyboards can lead to a large number of mistakes. The user who has an extended keyboard on the desk and a PowerBook keyboard on the road, normally finds the PowerBook keyboard difficult. The answer to reduced mistakes is Brian Bechtels' PB Keyboard Remapping resources. Brian provided three scripts for the Finder to allow it to remap the **Caps Lock** key, the **Escape** key and the **Tilde** key. In each case just drop the file on the system file in the system folder. Then use the keyboard Control Panel to select the new keyboard layout after rebooting. Once the keys are reconfigured, check out the file entitled H/W Hacking your PB keyboard on how to physically move the keycaps.

Powerbook Compatibility

7 § HW-PowerBook

The worst thing about a PowerBook is that it has batteries, and so the machine needs to slow down to save power. Testing programs on a PowerBook can be a real pain, since the battery problem is a real problem. To simulate the battery slowdown of the PowerBook, Brian Gaeke created PowerBook Compatibility. This hack makes a desktop Mac act like a PowerBook. The hard disk and the processor both run like a PowerBook on batteries, which makes debugging easier and faster.

Simulating batteries that are about to run out is cheaper than partially charging the batteries to get the same effect on the PowerBook.

PowerBook Pixels_ 1.0.1

7 § HW-PowerBook

PowerBook Pixels by Jon Wätte is a program that will find *shot pixels* on the active matrix PowerBooks. Shot pixels are dots on the screen that do not work anymore. With the active matrix screens, Apple allows five shot pixels before they will replace it free of charge. *Darker*, a part of PowerBook Pixels, darkens the whole screen so that the shot pixels will show up white. The other part of PowerBook Pixels is *Pixel Shootout*. Shootout can be used to simulate five random shot pixels so that programmers can see what their application will look like on a PowerBook with shot pixels.

Procedure Call Logger

7 § SW-MPW

Procedure Call Logger needs Macsbug and MPW to function. Marshall Clowe, the author, needed a way to track where he was in a program when debugging, so he rewrote the MPW profiling library to allow for user-defined calls. Using the BreakHere procedure, MPW determines whether or not to call Macsbug when trace on is chosen. This code can be modified to write a trace to a text file for future reference.

ProcessFinder

7

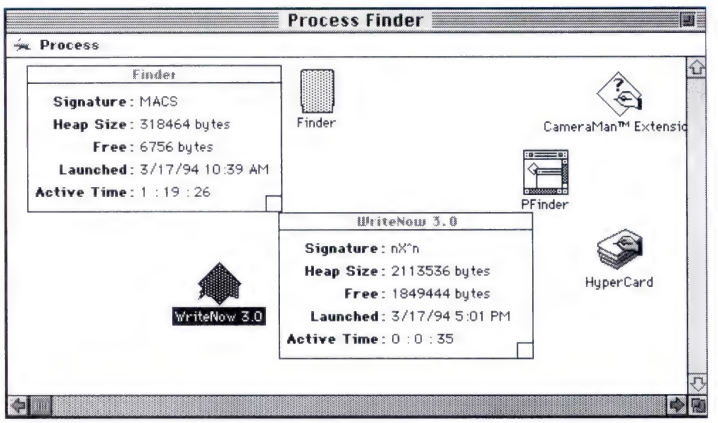


FIGURE 4.10 THE PROCESS FINDER WINDOW

Process Finder was written by Edward Harp from Pharos. Eric writes application frameworks for a living and wanted to show off just how complete they were, so he created an application that includes all of the **Finder** features in a window. The application includes a menu bar and draggable icons inside the Application window as can be seen in Figure 4.10. Notice that all of the applications that are running at the time have their own icons in the window. Also notice that the window has a menu bar. The menu bar includes the ability to quit any application that is running. Process Finder provides information on what the Signature of the application is (the creator name for the documents that it creates) the size of the heap zone that the application has, and how much of that heap is free and available to use). The heap information is updated as the memory usage changes. The launch time is also available, so that you can determine how long the application has run. Finally, the active time on the application is shown, so that you can determine what is using the majority of the processor time. This is very

useful when there are a number of things running and a limit to the memory each can have. By using Process Finder, memory limits for each program can quickly be explored.

Rapmaster

7

FIGURE 4.11 THE RAPMASTER BOOMBOX INTERFACE

Bryan K. "Beaker" Ressler is an engineer at Adobe. He is also an accomplished musician. Brian had the idea that a digital boombox would be a lot of fun to create. He built a full-working boombox with a rap scratch pad and everything. Additionally, he made it work with a microphone, so that voice overs are possible. Adding a few special effects and the beat and bass of most of the modern music types, Bryan built a great toy. He even included user help that is very complete and offers suggestions on how to use Rapmaster. Bryan built a complete application. By the way, Rapmaster's volume control goes to 11, not just 7. Did Jim Reekes have a hand in it, or did Bryan find a back door?

RISCy Bitsness

7 §

Paul Campbell, the genius behind Digital Film and the SuperMac Thunder cards and David Falkenberg, the driving force behind MacTCP, built a virtual RISC computer. They simulated a RISC machine that has fewer than 20 instructions and were able to get far enough along in the development process to get a smiling Mac face on the screen from a cold boot with the RISCy Bitsness installed as the operating system. The source code is included, but this hack is not finished because David went on to work on the kernel for the PowerMacs. All the work was done in a single day.

Run & Stumpy

7 §

Tom Lippincott is the primary author of Microphone and the author of Desktop Secretary. A long-time Mac user, Tom dreams of the days of 400k system disks. His idea was to at least allow a startup disk to be loaded from the floppy drive at boot up. Tom comments on Run and Stumpy:

"Slip a floppy with only a slim Startup Items folder into the drive, power on, and let your hard drive boot—in addition to your usual system, the items from the floppy are opened. Run is a program to put in the Startup Items folder of your hard drive. When you boot, it searches all volumes for Startup Items folders at the root level and opens the files within. Stumpy is to be applied to floppy drives. It installs stump boot blocks, which keep the disk from being ejected before the hard drive boots. It also makes a Startup Items folder and installs cute icons."

Savvy

7 §

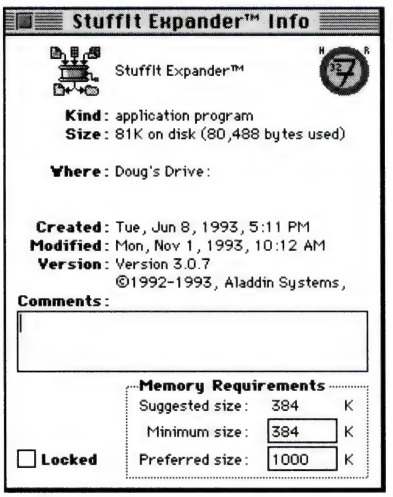


FIGURE 4.12 THE GET INFO BOX OF A SYSTEM 7 SAVVY PROGRAM

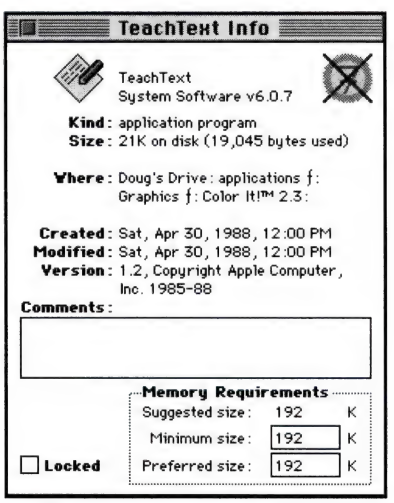


FIGURE 4.13 THE GET INFO BOX OF AN APPLICATION THAT IS NOT SYSTEM 7 SAVVY

Joel West writes drivers and low level code for a living. His applications and drivers have to work with everything. Joel found that there was a problem with programs that were not System 7 Savvy and his drivers. His solution was to write Savvy, an extension that graphically offers the user information on how savvy or not savvy their application is. If the program is 32-bit clean and will run with System 7 without problems the Get Info box will look like Figure 4.12. If the program is not 32-bit clean and will not run with System 7 without problems, the Get Info box will look like Figure 4.3 instead. Savvy provides an easy way to find out what does and does not run with System 7.

Scroll O Rama

7 §

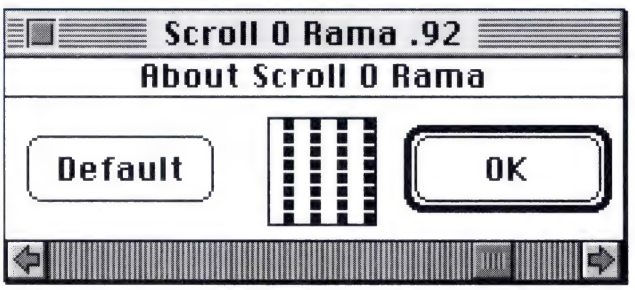


FIGURE 4.14 A SCROLL BAR AFTER SCROLL O RAMA IS DONE WITH IT

Jeff Walker runs his own software company while going to school at the University of Michigan. His hacks tend to be on the edge. Scroll O Rama banishes boring scroll bar patterns! Jeff indicated that this Control Panel is both dishwasher and microwave safe (though the Macintosh itself is not!). This is an example of a hack that is so well constructed that Jeff did not have to patch anything in the operating system to make it work.

Send the Hack

7 §

Demonstrating the power of AppleEvents, Todd Hunter created an extension that would make a menu of folders in the system folder, allow the user to select a folder, and then open the folder and all the files in it. That is all that Send the Hack does! If the Macintosh is a shared machine, Send the Hack can be used with folders filled with aliases to set up the machine with a single menu selection for each user. This is a demonstration hack, and the folder names are fixed, due to time limits. The source is included, and making the folder names dynamic as well as changing the location of the folders from the system folder to another location is an easy task for a good programmer.

Server Remote Control

7

As a network administrator, getting to the servers to restart them can be a problem. Server Remote Control allows a user to restart a server from a remote Macintosh. It has two parts, both of which must be installed for it to work. Server Controller has to be installed in the Startup folder of the system folder on the machine to be controlled. Then Remote Control allows the user to restart the machine that has Server Controller installed on it. Server Remote Control works with AppleShare 3.0 and System 7 File Sharing only. It was written by Jim Luther of DTS at Apple Computer.

ShowINIT Names

7

Do you wonder what the names are of the extensions that are loading? Do you watch the pretty icons when the Mac starts and then wonder what it is that the machine just did. Well, ShowINIT Names by Marvin Carlberg puts the names up with the icons during startup. This makes it easier to see which Control Panels also load as extensions. It has to load first to provide the names for the rest of the extensions.

SloppyCopy

7

Waiting for the Copying Files dialog to finish and go away can ruin your morning. Several companies have created commercial products to avoid this problem. Steve Falkenburg took advantage of AppleEvents and a system level patch to make SloppyCopy. It copies files in the background. File copying, then, does not stop the other work from going forward on the machine. Programs can be launched, and other file copies started. It is not as robust as the CopyDoubler product from Symantec, but it is very useful, if copying files makes the day longer now. SloppyCopy has problems with moving things into and out of the system folder and with remembering where icons should be in the windows.

Sound Asleep

7

David Thompson wrote Sound Asleep on a Macintosh IIci. This was a bit of a trick, because Sound Asleep will only work on the PowerBooks, Duos, and Mac Portable. This is an audio hack that noti-

fies the user when the portable is asleep and when it is awake. The hack was implemented as an application rather than an extension, to use it place it in the Startup folder of the system folder. The program was written in Assembler as the first hack that David did. It was finished in one day, without the aid of written documentation. Also, David had never used a portable before.

StickyClick 2.0

7 § 040

Sticky Click is a hack that adds the best feature of a trackball to the mouse. With a trackball, the mouse buttons can be sticky, that is, when a menu is chosen, the mouse button does not have to remain down to make the menu stay open. This is a short cut that allows the hand to relax and the user to be more productive. Steve Zeller, the author of FinderMenu, created this hack to provide the trackball functionality when using the mouse. Two versions are included: the very stable and well-tested version 1.2 and the MacHack version 2.0. Use 2.0 at your own risk, or better yet, download the latest version from an on-line service.

Strobe

7 HW-PowerBook

Turn your PowerBook into part of the lighting system in a disco. Barry Semo made it possible to flash the screen backlighting on the PowerBook off and on at a preset frequency. Since the backlighting on many of the PowerBooks do flash on and off to save the battery, Barry decided that controlling it with a numerical value, rather than a sliding control, is a better option. It can increase battery life by 10 to 15%. This frequency can be changed in the control panel. Barry claimed that the next version would do dots and dashes and be useful for sending morse code.

SwapMenus

6 7

Pete Helme is one of the first Macintosh DTS engineers. SwapMenus was born of one of the best motivations that a programmer can have: laziness. Pete did not like to have to move from one side of a large monitor to the other to get to menus. Swap Menus allows the menus to be dragged from one location in the menu bar to another and dropped there. This is very handy when someone rearranges the menu order in an application.

Text Capture FKEY

7 §

In many programs, when a block of text is highlighted and copied to the clipboard, the text is not text but a graphic file, like a PICT file. This means that the text cannot be edited, only viewed and placed. Text Capture FKEY changes the nature of what is copied to the clipboard with most applications. If the program is drawing the text on the screen as text, then the clipboard will contain only the styled text. If the program is drawing the text on the screen as a bitmap (like MacPaint), then the copy on the clipboard will still be a graphic. This is a quick way to get text out of windows like Modal dialog boxes in editable form. James W. Walker wrote Text Capture FKEY.

The Mac Clapper

7 HW-Sound Input Device

With a microphone, the MacClapper by Iggi Monahelis will turn off the Mac. This is a small application that can be left running in the background. With older Macs (Macintosh IIfx and earlier!), the MacClapper will just open, make a clapping sound twice, and then

quit. With Macintosh IIsi and newer, if the microphone is plugged in, it will open and wait for someone to clap their hands to shut down the machine. Using this with the television running on a show with an audience can be very frustrating; every time the audience claps and you look to see why, when you look back the Mac is off.

The Regulator 1.2

7

Brian Bechtel wrote a new version of the Regulator for MacHack and continues to update the program from time to time. This extension allows a user to set up the PowerBook or Duo hard disk for optimal use with batteries and then switch to the maximum settings when the portable is plugged into the wall. The setup parts that The Regulator effects are the System Sleep Time and the Hard Disk Sleep Time.

Trash Selector

7 §

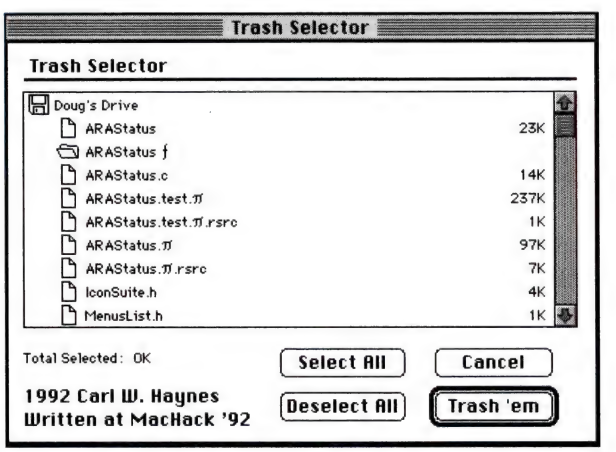


FIGURE 4.15 THE TRASH SELECTOR WINDOW

Ever throw something away by accident? Trash Selector by Carl Haynes allows the user to choose the items they want to throw away. When empty trash is selected from the menu, Trash Selector, displays a list of the items in the trash so that choices can be made on what to throw away from the list, as shown in Figure 4.15.

Ümlâût Õmêléttè

7

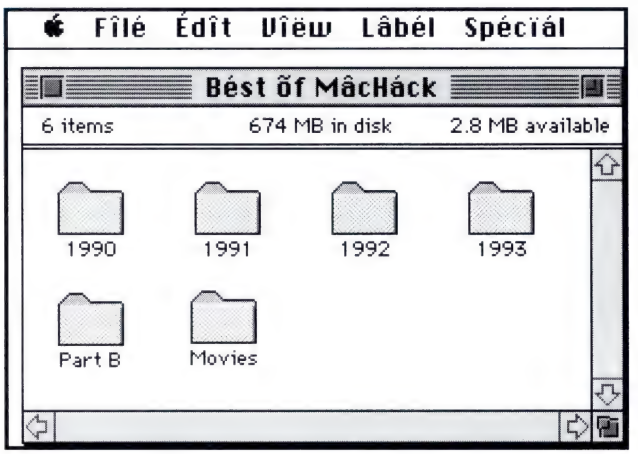


FIGURE 4.16 FINDER WINDOWS WITH ÜMLÂÛT ÕMÊLÉTTÈ

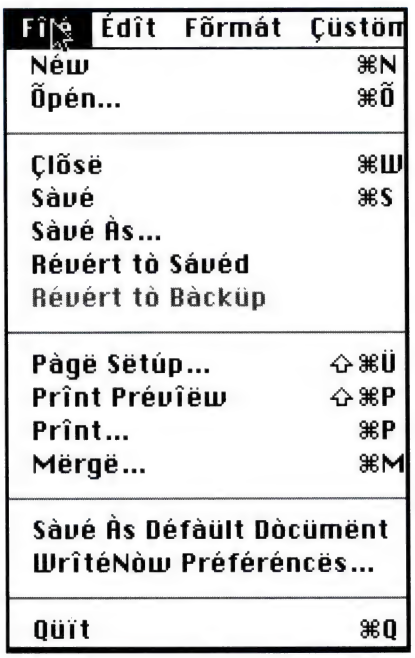


FIGURE 4.17 MENUS WITH ÜMLÅÛT ÕMÊLÉTTÈ

Tim Dierks thought that Diacriticals were lots of fun to deal with, hence Ümlåût Õmêléttè. This extension provides them in all of the menus and window titles. It is a great practical joke.

Windows 3.1

7

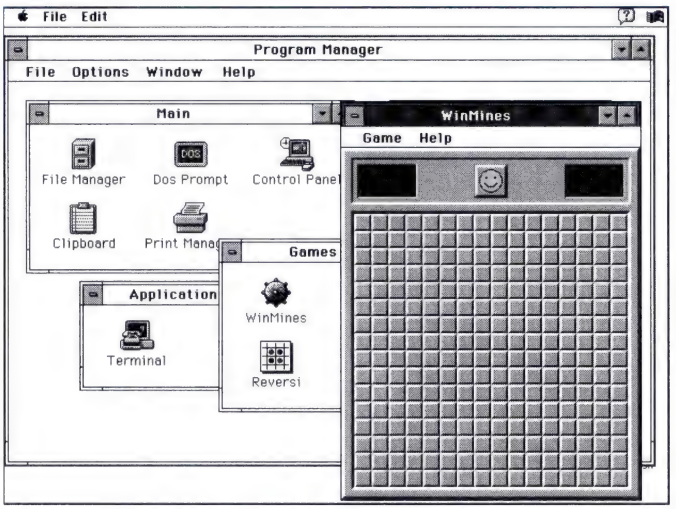


FIGURE 4.18 WINMINES ON THE MACINTOSH, OH NO!

Bob Spence created what appears to be a full windows interface for the Macintosh. Complete with the games that Microsoft ships with Windows, the only application that will really run is the WinMines game. The rest of the icons produce the correct Windows error message for not launching: from an invalid path for the file manager to a com port problem in terminal. Each error message shows up with the correct font and style of window, and the error messages are all plausible. This hack can be lots of fun when used on a Mac in the office. Ask for help with the configuration from the local windows guru and watch the fun begin.

CHAPTER 5

1993—SegaMac and the PowerPC

MacHack 1992 ended with a meeting to determine who the 1993 keynote speaker would be. The group of attendees that attended the midnight planning meeting for 1993 chose Bill Gates from Microsoft as the first choice and then eight more names as alternates.

In the meantime, the press had taken the information that had been presented at MacHack and published a number of articles. *MacWeek* ran several articles detailing technology from MacHack 1992 and talking about the Hack contest. Raines Cohen and Steven Howard took several weeks to tell all the tales that they heard at MacHack. It took longer for the material to appear in *MacUser*, but Stephen Somogyi provided a detailed look at much of the technology that was unveiled at MacHack, without mentioning MacHack in the article. The press discussed the fallout from MacHack for several months; again, many of the stories did not directly reference MacHack.

It was not until late October that the "Thanks for asking but no thank you" note from Mr. Gate's assistant arrived. In the next six months, all eight of the alternates also declined. Several because they had just made plans for the time around MacHack. Without a keynote, there was no conference framework. Hence, MacHack proceeded very slowly. In fact, the committee fell further and further behind. Finally in late

March, someone suggested going without a keynote. The suggestion carried, and the committee moved forward in high gear.

Apple had taken the feedback from MacHack '92 and offered lots of help for 1993. As a result, a delegation from MacHack visited the Apple Campus in Cupertino in early January with a set of meeting already arranged. Every mover and shaker in the software part of Apple was on the list. As the time for each meeting approached, one delegate would call the office of the person with whom they were to meet and request someone to escort them upstairs. The appointments were suddenly and mysteriously canceled. Instead of being met at the door in City Center for lunch with Kirk Loevner, the delegation found itself sitting in the courtyard trying to catch people as they went to the cafeteria for lunch. Several of the DTS engineers shared sandwiches and soda. If they had not done this, no one in the delegation would have had any lunch.

There were news vans at the main building, almost as if there was a major crime that had been committed on campus. Kirk Loevner and others were seen dashing about the campus as they attempted damage control with the press. The delegation went back to San Francisco that evening without a clue as to why the day had gone so poorly. The San Francisco stations carried interviews that evening. Apple's VP of Software Development, Roger Heinen, was skipping town and joining the number one rival, Microsoft. Headlines in the San Jose Mercury News the next morning confirmed the report. Now the reason for so many canceled meetings became clear.

Steve Weyl was the one bright spot in that otherwise dismal visit, in contrast to the rest of the chaotic day. Steve offered ten technical sessions, two of which were mystery sessions. Because Apple had not yet made an announcement to the world, MacHack could not publish what the sessions were. Steve was very supportive of MacHack and its goals. He had learned a number of lessons at MacHack '92 and was trying to apply them internally at Apple. The chaos at Apple in early 1993 was tremendous. Yet Steve Wyle seemed to ride above the storm, keeping his troops working and focused on the software.

MacWorld San Francisco opened two days later, and the news was all over the show floor, the key manager at Apple for the future was

gone. "Apple's strategy was sold out to Microsoft" and other such comments were heard from a number of sources. The move even caught *MacWeek's* infamous "Knife" flat-footed.

The Knife has been a regular pain in Apple's side for years. Apple's internal security group pursued people they suspected of giving information to the Knife. Apple even refused to advertise in *MacWeek* for a couple of years because of the Knife and the *MacWeek* attitude, which was "if they could find the information, then they should publish it." Raines Cohen had more than once been refused entrance to WWDC because of his affiliation with *MacWeek*, even though he was affiliated with an Apple certified developer. Steven Howard, on the other hand, had several times been admitted to WWDC while working for *MacWeek*. The Knife has been the recipient of information from many of the industries' best engineers. They read an article or press release that they know is wrong and try to correct it by calling the Knife. *MacWeek* uses the Knife column to present anonymous rumors. The engineers use the Knife to lobby for causes that they believe in. The whole Macintosh community benefits from the column. The Knife's true identity is a matter of speculation.

Given the confusion at Apple and the departure of a number of key executives and engineers the mood surrounding WWDC was bleak. Everyone expected that the Developer's conference would be full of half-finished software. The air was full of defeat. It was the quietest MacWorld on record. Everyone was moping around wondering if there was a future for the Macintosh.

From mid-March to May the MacHack schedule started to come together. It was a mishmash of old and new. The idea of staggered sessions and around the clock scheduling from 1992 stayed. The need for more content was reinforced. The result was that over 70 technical sessions slots were scheduled for 1993, the most ever for a MacHack. This meant that the committee had to dig hard to find enough material to fill all the slots and more importantly, that the material and speakers had to be very good. Thanks to Steve Wyle and Rick Fleischman this turned out not to be a real trick.

Jordan Mattson had moved over to the PowerPC team and was interested in a strong showing for PowerPC at MacHack. He indicated

that there would be a need for people to sign nondisclosure forms in order to get into part of the machine room and into several of the sessions. This led to the planning of a third machine room. The Apple PIE team lead by Mike Engbar wanted a chance to show off Apple's PDA, the Newton to the attendees and to explain what it was all about. Darin Adler who had left Apple to join General Magic also required nondisclosures for a late-night session on what became TeleScript. All of these sessions and nondisclosures changed the flavor of MacHack. The Hack contest was de-emphasized, and sessions were the major attraction.

May brought the Developer's conference and the second annual "Take Brita to Great America" trip. About 30 MacHackers arrived at the Holiday Inn lobby to go to Great America. This group of midwest and eastern nerds had a great time analyzing the physics of the roller coasters and attempting to see who was the largest programmer that could fit on a number of the rides.

At the log flume, four of the largest programmers squeezed into one of the logs. The bet from the others was that the log would sink as soon as it left the dock. It rode on its wheels most of the way around the flume, so they may have been right. The log took the steep drop off the top of the hill of the log flume and plunged into the pool. It sent a wave of water shooting over the walkway and into the waiting crowd. Several bystanders clapped, and the ride host indicated that he had never seen a wall of water quite that high before. The four programmers in the log exceeded 1200 pounds by a good bit! While on the log flume with two Apple engineer's, it became apparent to the other riders that they were doing a session on a piece of the operating system that was not on the MacHack schedule. This was changed shortly after the log took a steep drop to the pool.

This happy interlude at that Great America more than compensated for the lost day in January. Several Apple engineers introduced themselves and offered to speak at MacHack. There was never a better office than the Top Gun roller coaster at Great America!

WWDC started on Monday with passing out CD-ROMs filled with code. Having listened at the gripe session in 1992, Steve Wyle provided a CD-ROM filled with new software. The disk was called

"Apple New Technologies." There were several pieces of code on the disk that had never been available to the programmers at WWDC. For programmers, the disk held the Apple Shared Library Manager, a way to share code between programs. For nonprogrammers, read marketing types, there was information on the Apple market around the world, one of the first times that Apple had shared real numbers with their partners.

For the first time, Apple acknowledged the importance of the Macintosh as a game machine with its publication *The Mac Game Developer's Handbook*. New versions of QuickTime and QuickTime for Windows were included; this made QuickTime the first cross-platform video standard. The rest of the disk was packed with other technology that will see daylight in System 7.5. Apple offered this as a way to say that they goofed in 1992. In fact, the speaker said exactly that when the disk was made available.

Several managers at Apple privately acknowledged that the feedback session at MacHack in 1992 led to the development of the New Technologies disk. They also acknowledged that Apple was paying more attention to the small developers because of what they saw at MacHack in 1992 and the feedback that continued to pour into the Apple engineering offices from MacHack attendees. Apple could not have started off the conference in a better light.

Through the course of the week, Apple offered looks at a number of significant new pieces of the Macintosh Operating System. WWDC turned out for the first time in three years to have more content than the attendees could possibly absorb.

Unfortunately for most attendees, the real future of the operating system was hidden in a session called "New System Software Extensions." This session had three new technologies to cover in just over an hour. True multitasking—making a computer deal correctly with two or more applications at the same time was presented in less than 20 minutes. The session competed with five others and was not described very well in the program; very few people attended. The Tread Manager, the basis of the true multitasking, was very well received and demonstrated by Eric Anderson, its designer, using a program called Traffic Manager. Traffic Manager is a cute simulation of a traffic light

at an intersection. The Traffic Manager simulation showed how the user of the computer could take control of the computer and direct for the first time what percentage of the computer's power was dedicated to the modem, the printer, the game that they were playing, and the rendering application that was running in the background. True multitasking is the major feature of UNIX. A file system that did not have the limits of the current Apple File System was also presented in this session. This file system broke the limits for the number of files that could be placed on a single volume, making it possible to fully use a 1-gigabyte hard disk. The current file system was designed for use with 20-megabyte hard disks. The new file system will allow users to once again make full use of their computers, as did the switch in 1987 from the original Macintosh File System to the current HFS.

Both the presenters were invited to MacHack on the spot and both accepted. WWDC ended with a hippy 1960s lovefest, complete with love beads; it spoke well of the feelings that most developers felt at the end of the conference.

MacHack '93

There were four weeks to make the schedule changes that WWDC brought about. Those four weeks ended up with weekly conference calls coordinated by Rick Fleischman and Chris Allen, the conference chair. Dan Repko and a team from the University of Michigan's Computer Aided Engineering Network (CAEN) took a tour of the hotel a week before the conference and made sure that they were comfortable with the way things looked. APS shipped 10 gigabytes of disk to the committee and Apple shipped all of the latest CD-ROMs. Dantz Development sent enough Retrospect Remote copies that the whole network could be backed up. Tribe sent a TribeStar to handle the PowerBooks. Apple corporate sent several boxes that were not to be opened by anyone other than Jordan Mattson. Hewlett Packard sent out two high-end color printers, and SuperMac sent out several large monitors. For the first time, the sponsorship of MacHack was complete in every way. The local Apple office in the guise of Steve King sent some of the top hardware that Apple was then shipping. The

machine room overflowed into three rooms, and it could have used a fourth. This was before the people who were carrying PowerBooks arrived. The team from CAEN, were ready to set up at 9 AM. They had three vans full of Macintoshes and printers. It took just over four hours for four people to connect and configure over 50 machines. There were people carrying in machines, people setting them on the tables and connecting mice, keyboards, and monitors. There was one person who was running the networking cable from machine to machine. The hotel engineer was running back and forth, checking that the power had not blown out yet. This when the machines were not yet plugged into the power system. One person crawled under the tables, ducking under the table clothes, and passed cords up from under the tables to the waiting people above. A waitress came in to see if anyone needed anything to eat or drink and then screamed. She thought that Thing from the Adam's Family was loose in the room when only the hand of the person under that table showed from behind the machine. Everyone had a great laugh when the reason for the scream was explained and the person under the table ducked up and said, "hello." Boxes of wires and cords were coming into the room and disappearing into the assembly. The network was complete in each room, and the wire was taped to the top of the door frames to allow the network to be continuous between the machine rooms. Ceiling tiles were lifted to allow for the network to run across the hall and out into the Holidome around the pool. The hotel engineer was worried that all the computer equipment would blow the circuit breakers in that end of the hotel; in fact, it once did. But once the system was up and running it was rock stable.

Jordan arrived and sealed himself in the third room that formed the machine area. No one was allowed in or out. Jordan would appear at the door and ask for a cable or a keyboard to be passed in. This disembodied head was frequently seen floating in the hallway as Jordan attempted to keep the door to the room closed as long as possible. In some ways, the setup time felt more like staging a Halloween play than setting up the equivalent of a medium-sized commercial network. There were problems with parts of the equipment that was rented, but the rental repair person could not get in to fix it at first. The poor repair person spent almost an hour standing in the hallway trying to figure out how to fix the machine without going into the room, or the machine having to leave the room. Jordan finally allowed the person

into the room. The engineer that Jordan had left guarding the room did not understand that the repair person was ok. Finally, Jordan had his part of the machine room ready. Jordan carried the only key, and the room was only open when Jordan was available. Everyone wonders why Jordan rented a room at the hotel, because the special machine room was open almost 24 hours a day. Once again, Jordan did not appear to most people to sleep during all of MacHack. Everyone was beginning to believe that Jordan is really twins since he always seems to be in two places at once.

As the machine room was being set up, there was a party to welcome the speakers going on at Dave Koeiol's. In reality, the party is there so that the machine room team can finish setting up the machine room, while people are arriving. If there was not a speaker's party the line to get into the machine room would form on Monday afternoon and loop the building by the time the machine room opened on Tuesday late evening. The "Welcome to MacHack" session that replaced the keynote was held one minute after midnight on Tuesday. This gave the folks holding the speaker party an end time. Normally the speaker's party keeps going until the last of the food and drink is consumed.

Chris Allen and Carol Lynn welcomed people and were followed by Greg Marriott and Marshall Clow who explained the Hack contest rules and how to create hacks. At 2 AM Greg Marriott spent two hours covering the keys to great hacks and how to make the most of the time at MacHack. The session is an annual rite and is called the "Essence of the Hack." The room was packed, even though it was 2 AM. The room was still packed at 4 AM when Greg finished. Most of the people migrated to the machine room, rather than to their rooms to sleep. It was not until 6 AM, as the sun began to creep down that hallway, that the machine room finally started to empty.

Wednesday's first sessions started at 8 AM, a scant two hours after most of the attendees had headed for bed. It was full, but most folks had the same attitude that they had at 8 AM classes—"do I have to" and "is this for real" were the operative questions people had. By mid-morning, the topics of QuickDrawGX and Multimedia were both addressed in formal sessions.

The first paper of the day was Shane Looker's *Parallel Processing on a Macintosh Network*. Shane offered a look at how a local area network could be used as a single computer to handle very complex problems. The complex problem would use the other computers on the network as a background process, and make use of otherwise unused cycles. Gary Odom's *Comparing C-Based Object Systems* followed Shane's paper. Gary dealt with the differences between C++, Objective C, and OOPC. His discussion made points for the C++ architecture, which seems to be the new programming paradigm for the Macintosh.

Rick Fleischman of Apple and David Neal from Symantec offered the most popular session of the morning: a look at Bedrock, Apple's cross-platform framework that was in development with the help of Symantec. This session was heavily attended by programmers, some of whom were not at all interested in what was being said, but rather wanted to voice options on the death of MacApp. Rick carried much of what was said back to Apple, which has resulted in the resurrection of MacApp.

Richard Clark offered the first of several sessions on PowerPC. This session was also heavily attended. Richard discussed programming for both the PowerPC and the 68K-based traditional Macintoshes. Richard also spent time discussing the architecture of the PowerPC hardware. Jordan Mattson came in at the close of the session and passed out nondisclosures (NDA) for the "secret machine room." People were surprised that Apple was willing to share with them a technology that was almost a year from shipping at MacHack. Only one attendee had a negative reaction to the NDA. There was steady traffic into Jordan's room for the rest of the week. Everyone wanted to try their programs on the PowerPC. The results were surprising: almost everything that anyone tried worked on the PowerPC without needing any changes. This machine was a year away from shipping and was rock stable.

Jon Wätte from Sweden finished off the morning with a paper on localization titled (*Swedish to You is like Greek to Me*). It is a look at how hard it is for someone who is not a native speaker of a language to get the dialog boxes and menus correct. Jon drew a crowd for lunch at his

table and like all the presenters of papers, found that there were always people who wanted a little of his time whenever he was not programming.

Don Brown, the founding programmer of CE Software, took a good look at the QuickMail Applications Programming Interfaces for the start of the afternoon. This new architecture of QuickMail allows programmers to add QuickMail directly into their applications and databases. Don also added comments on network hacking and answered more than a few questions on another of CE's products, Quickkeys. As always, Don presented the information in a lively and upbeat manner and was able to zoom in on the questions and to answer them clearly.

Marshall Clow and Dan Lipton tackled printing using QuickDrawGX. Marshall is the senior programmer at H-P for the deskwriters and Dan is Apple's senior engineer on the GX printing package. They were able to offer an in-depth look at how GX will change the way that printers will work in the future; from implementing Leonard's Desktop Printers to combining Postscript and Quickdraw in a single model. The use of TrueType and Postscript fonts interchangeably is one of the nicest features of GX. Printing with GX will offer users more control over how the documents print. From better gray scale pictures to better toner control for draft documents.

Meanwhile, hidden away upstairs was Leonard Rosenthal, taking apart code that programmers were having trouble with and helping them to understand what is wrong and how to fix it.

Some people feel that the code clinics are the covert part of MacHack. Leonard and his friends spend the afternoon diagnosing problems and helping others. Many of the first time MacHack attendees don't even realize that the code clinics are available. Leonard is supported in his code clinics by almost all of the MacHack veterans, they all pitch in to help new programmers understand the problem that they are having and to help them overcome the hurdle. As the Macintosh has become more complex over the years, the code clinics have become more critical. Leonard, Marshall, and others were wearing SmartFriend™ t-shirts during most of the conference. Leonard's code clinics are the best way to become part of the SmartFriend network.

"Core Tools on the Macintosh" was Rick Fleischman second session of the day. For the programmers who had not been at WWDC, it was their first chance to hear that Apple was backing away from Macintosh Programmer's Workshop (MPW), the basis of all of Apple's programming tools, and moving to Symantec C and C++. This revelation caused more than a few of the programmers to shake their heads in frustration, and the questions were frequently angry. Even though most of the programmers felt that Symantec had superior tools, the feeling was that MPW was needed as a whipping boy for Symantec to continue to improve their software tools. Apple had made development tools a profit center, just like they had done with Developer programs a few years before. The tools group now had to sell enough tools to make a profit after salaries and overhead were paid. This meant that unless MPW and the other tools sales picked up significantly, the tools group could not continue to devote the resources to MPW that were required to make it a viable alternative. The fact that Symantec needed a whipping boy was borne out in many people's minds by the buggy 6.0 release of Symantec C and C++.

David Feldman, a former Blue Meanie and multimedia tool developer joined the ranks of the suits. He gave a talk on the uses and abuses of Venture capital in software development. His talk was well attended and did not have the controversy that the other talks in the early afternoon generated.

The final speaker of the afternoon was Steve Falkenberg, a University of Michigan and MacHack alumni, discussing Apple Open Collaborative Environment (AOCE). AOCE or "ouchie," as it is known inside the development team, offers for the first time a way for programs to work together to operate on the same document. This system of calls based on AppleEvents and Applescript allows programmers to call each other's program and to use the tools that are needed from the other application rather then coding it all over again. Also, AOCE allows several people to work on a document at the same time. This flexibility requires that applications be written to a specification that Apple has published. If the application is written correctly, then it can be part of the AOCE system, and if not, then it cannot be part of the system. Steve's job at Apple was to help programmers make their

applications AOCE savvy. Steve's talk offered lots of practical advice and code samples for people to take home.

The dinner hour was extended because the audio visual system was not quite ready for the Newton session. The Newton refused to be projected clearly on the large screen. Finally, the Newton Session was postponed until after the Bash Apple session.

The Bash Apple session focused on tools and the lack of Apple tools for the future. At the Bash Apple session there were easily 40 people on the Apple side of the table. There were 20 or so people in the audience that used to sit on the Apple side of the table. Several attendees were sitting on the other side of the table for the third or fourth time. Darin Adler was told at the beginning of the session that he might not get answers to all his questions. After all, Darin had been the leader of the group that most of the Apple people were part of only a few weeks before, and Darin had helped set most of the direction. In many cases Darin knew more about programs than the people who were assigned to work on them. There are comments in Washington about revolving doors. Apple also seems to have developed a revolving door for engineers. It also focused on changes in the developer programs and the lack of prototype hardware for developers. For small developers, in the past, Apple had offered extra help, so that they would have an advantage in the market. This was the difference that allowed Aldus to launch PageMaker and many other companies to launch successful products. With Apple focused on the large developers, there was no chance that there would be another small company that would have an early breakthrough product. Apple was again focused on the question of how many machines the developer sold for them yesterday, rather than the question that they should have been asking, which was "how many machines can you sell for me next year?". All the information that was conveyed at the session was again recorded and taken back to Apple. Many of the items that were questions or comments at the Bash Apple session have changed since, many in a way that was suggested at the Bash Apple session. Although Steve Weyl was not there in person, he had designated two people to take notes and report back. Because the policy-makers were absent, the Bash Apple session was one of positive question and answers with comments to be taken back to higher management. The key issues were AppleLinked (sent by electronic

mail) to Apple that night, and the responses to several were posted late Thursday night.

Bash Apple ran until midnight and ended with a Nerf fight, Nerf bows and arrows versus all sorts of other Nerf toys. This good-natured fun replaced the water and silly string fights from years' past.

The room was returned to a state of order by midnight, and Mike Engbar and Tony Espinoza from ApplePIE took the stage. They discussed the Newton and its development system in detail. The session ran for over two hours, and the ballroom was packed. There were more people at that session than there were at the Bash Apple session, traditionally the full-house session. Mike and Tony received the first standing ovation of the weekend, and that made headlines in the following week's *MacWeek*. Raines Cohen wrote more about that one session than he did the whole rest of MacHack.

Following Tony and Mike was an impromptu session by Darin Adler on General Magic and MagicCap. Darin had been asked at the Bash Apple session by several of the Apple people to show off what he was working on. Darin asked if it was alright to add a session on the fly. The session started at 2 AM, yet the room, was packed as usual. Darin finished this look into the world of agents in the wee hours of the morning and received the second standing ovation of the weekend. Many hacks did not get finished at MacHack because of these two sessions. No one really cared that the hacks were not finished. They had a pocket-full of the future and the mood was upbeat. The manager of the bar complained for the first time in years. His business was way off that evening. Not that the business was ever great during MacHack, but this year people were not even stopping in for coffee and Coke; they were too busy.

Greg Marriott was in the machine room answering questions with the help of Eric Shapiro and others until the sun again rose and began to fill the hallways. Sometimes people ask if MacHack is a vampire convention because of the hours that are kept. The housekeeping teams cannot start until noon. And they are on duty until 10 PM, so that late sleepers can get the towels changed when they wake up. The Holidome lights are on all night each night, and the room service and catering teams offer food until 5 AM. When the coffee was not refreshed

at 3 AM, it was a major crisis on Wednesday night (Thursday morning). Most of the attendees had not been to sleep in 48 hours, and they were running on straight caffeine. The complaint from one attendee was that there was too much blood in his caffeine system! One programmer nodded off in the middle of typing because there was not caffeine. Carol Lynn, the conference manager solved the problem within an hour.

Keith Stattenfield of Apple opened the program the next morning with a session on AOCE and gateways. AOCE is designed to work over a network as well as on a single machine. More importantly, AOCE is designed to work on machines that are not Macintoshes. The use of gateways allows the software to escape the machine that it starts on and operate on the target machines. Keith is the DTS engineer that is charged with making sure people understand how gateways should be accessed and used. This session looked into the heart of AOCE and how it would be able to span not only applications, but machines and networks. Maurita Plouff offered an interesting paper that has had impact at General Magic and other places called *"Images of Ourselves: The Electronic Little People,"* which questioned whether agents should take on the characteristics of their owners, the facial features and expressions as a beginning and more of the personality of the owner as the software matures, including the ability to mimic the owner's emotions and moods. This has resulted in a number of discussions on the various networks and changes to the way that some companies are positioning their future software. It has even changed in many people's minds the meaning of the word "agent."

John Clark offered a look at development tools and what it was going to take to get to the tools of the future. John's approach won him a job offer on the spot from one firm, but he chose to stay in Chicago instead. John's look at the cost of tools that are inadequate and the need to define and develop better tools harkens to the future of all operating systems. As they become more complex and users ask the computers to do more, the quality of the tools will determine which machines make it into the market for people to use.

Eric Shapiro offered tips on using multimedia and video in applications, and Bill Fernandez offered a look at what users see as speed in an application and how slowing down the software's real function can

make the software look like it is really faster than it is. Some of these tricks included zooming rectangles when resizing windows and beeps, color changes, and so on, which would make it look like the application is screaming along, instead of crawling.

The afternoon opened with a paper by Bernard Bernstein and Chris DiGiano. They looked at debugging by listening to the sounds that the code made as it executed. Each section of code would have a different sound, and that sound would allow the programmer to sonically debug the code. This paper resulted in one of the silliest hacks at MacHack, "a sonic compiler." The concept of allowing the programmer to track the code that is executing without having to break away from the application is a "sound one," and at least one company is pursuing the idea.

Alma Whitten and Robert McCartney offered one of the most ambitious papers at the conference. They tackled a voice recognition toolbox that could be used with any application by any user. This set of tools built in Macintosh Common LISP is a complex topic that they each presented very well at MacHack. The idea is that by using AppleEvents, the user can drive the Macintosh with a microphone. Since a model of speech is not needed for each user, many people can use one machine without training the machine. The idea that the user can speak plain English and the computer will translate it to computer speak means that the user also does not have to be trained. By looking for keywords on the fly, the software offers the first true natural language interface for computing. This has been the goal of many computer makers for years, and it offers the hope that most people will be able to program their VCRs and Microwaves someday. The key to software is the use of AppleEvents. With the advent of AppleScript, a single keyword offers the ability for the computer to hear "graph yesterday's sales data." It then hears: dial the mainframe, download the sales information, pull the information into a spreadsheet, format it, open a graph window, and finally, create a graph complete with headings and proper formatting. It might even know how to compare information from week to week, month to month, and so on. The complete toolkit was offered to the attendees and is included on the CD in this book.

David Falkenberg tackled the Thread Manager for Eric Anderson, a little-known extension of the Macintosh that allows multitasking,

that is, true cooperative multitasking like UNIX has. Dave offered a glimpse of the MicroKernel, the basis of System 8, and what the future for the Macintosh might hold when the PowerPC is available.

Richard Clark followed with a session on porting code to the PowerPC, and then Paul Campbell took up the PCI bus, the future bus for the PowerMacs. The day was packed with full-house sessions. This had an effect on the number of hacks that were ready for the Hack contest—not much, but enough to allow the Hack contest to finish before dawn.

Dr. Steven Lewis in his paper presentation, dealt with the idea of portable code libraries: that is, making it possible to write a program once and compile it on several machines. The key is libraries that are optimized for each machine, so that the programmer can concentrate on making the program as useful as possible to the people who buy it. This is a simple concept, but a number of companies are still chasing it.

Gary Kacmarcik discussed the PowerPC. His paper examined the changes in the way that the lowest levels of the programs will have to take advantage of the RISC processor. He offered a tool that let programmers try their PowerPC code on the Macintosh with a 68040 processor in it, sort of a PowerPC emulator for the Macintosh. Gary delved into a number of key areas and discussed how programmers would have to change the way that they are doing things now to make the programs efficient on the PowerPC.

After the papers were finished, the most common sight in the hotel was the pizza delivery person, everyone was either in the machine room hacking, in the Holidome, or their rooms. The network was jammed as people tried to enter hacks. Greg and Scott Boyd were in the ballroom setting up for the Hack contest. They did it a little differently this time. Instead of a single machine that everyone would use, they set-up three stations, each with a large screen projector and a 15-foot diagonal screen. All three were wired for sound and microphones were at each station. An Apple DTS staff person was in charge of each station. In other words, somehow, they were intent on keeping the Hack contest running. Midnight rolled around, and the thunderstorm did not. The room was packed with people—over 300 people

were in the room when the Hack contest started. Many of them were not members of MacHack, just folks who wanted to be part of history. Please note, the hacks were not presented in the order that follows.

The 1993 Hacks

AppBar by Donald Brown

7 §

Don Brown is the creator of CE's Software's Quickkeys and many other Macintosh applications. Like many programmers, Don has several applications open on a machine with several monitors. To get to the menu bar to change the application that is in front is a real pain. Don created AppBar, a floating window that always stays in front. It grows and contracts as the number of applications that are running increases and decreases. AppBar can be reduced to just the title area to save screen space. By using a combination of AppBar and hide windows from the finder, it is possible to keep the screen relatively clean and unencumbered. Once started, restarting the Mac is the only way to quit from AppBar, which is an application and does not patch any traps. This is one of three hacks that Don created using some neat tricks that he discovered in the window manager. This application took less than three hours to build.

AppleScript™ Worm Folder by Josh Goldfoot

7 SW-AppleScript 7

Josh wrote the first AppleScript virus, or so that is the way that he introduced it to the audience. The worm is a self-replicating AppleScript application that will make copies of itself in every volume that it sees. This was done not to show how to be malicious, but rather to show

how software distribution can be done with AppleScript. It also shows the power of the AppleScript language. John Norstad, the author of Disinfectant, spent the rest of the night figuring out how to defend against this kind of virus, should anyone ever try it. The AppleScript Worm on the disk will replicate, but it is not harmful.

ChooserHack by Nick Kledzik

7 §

Nick, like many people on a large network, found that the chooser was a poor way to get to servers, so he created an extension that replaces the chooser with a folder on the desktop. This folder contains a folder for each zone on the network. In each zone folder is a folder for each server, and in the server folders are aliases to each of the volumes on the servers. This sounds complex until you realize that the key aliases can be placed in a single new folder or in the start-up items folder in the system folder. To reduce the time and network traffic that it takes to create the folders, Nick only builds the **Zone** folders on the first pass. The user has to open the **Zone** folder before the **Server** folders are created; in addition, the same is true of the volume aliases. Finally, he does not actually build the volume alias until the user tries it the first time, using a shell of an alias until the user actually mounts the volume for the first time. All of this saves traffic and allows the folder to take the minimum amount of disk space, on a user's machine. There was wild applause when people realized the power of this hack.

ClipShare by Brian Topping

7

Sometimes an item that is on the clipboard needs to be on another Macintosh. There is no easy way to share the clipboard between two or more Macintoshes. Brian solved this problem with ClipShare. The application allows a user to share the clipboard between two or more machines over the network; no more making a file and then sending it by QuickMail when a clipboard will do.

CloakShare by Eric Traut and Dan Clifford

7 §

Running personal file sharing on a machine can mean that a number of people halfway across the company can find some inappropriate things. So Eric and Dan wrote CloakShare, which makes a machine that is running personal filesharing and cloakshare invisible to everyone who is not running CloakShare. The machine can be shared only by those people who are intended to share it, even if the security is not set up correctly. This hack was run across all three of the machines to show that the man in the middle could not find the file server if he did not have CloakShare installed. Since the DTS engineer running the middle machine was one of the managers of DTS, his troops in the audience cheered when the file server did not show up. They thought that it would be cool to hide the disk from the boss.

Clock Menu by Francis Stanbach

7

FIGURE 5.1 CLOCK MENU'S BOOT TIME ICON

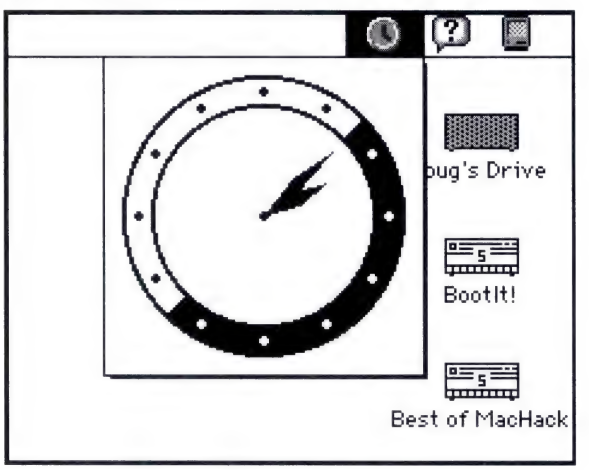


FIGURE 5.2 CLOCK MENU INSTALLED

Francis, the original author of Apple's Teach Text, built a Control Panel to add a clock menu to the finder. The neat part of Clock Menu is not the clock or the menu, it is the startup icon. As the icons roll across the bottom of the screen (see Figure 5.1), Clock Menu throws up an icon that is 8 times larger than the normal icon. It makes a visual statement about the fact that it is loading. The analog clock shown in Figure 5.2 is the rest of the hack.

ConvertToMovie Jr.

7 §

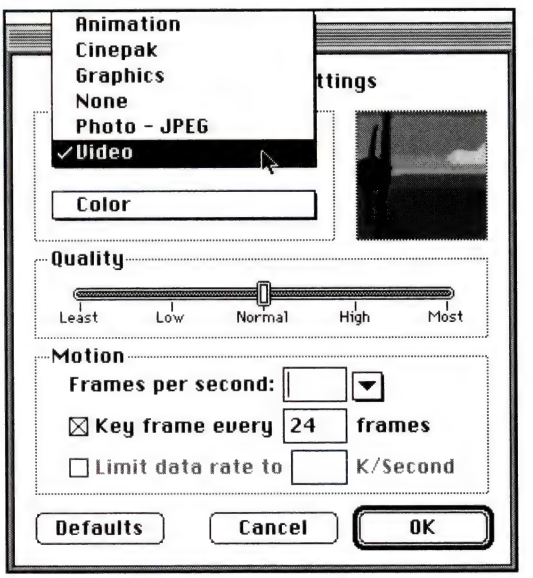


FIGURE 5.3 THE CONTROLS FOR CONVERT TO MOVIE JR.

When a QuickTime movie is made, frequently, it is not optimized for a particular situation, like making it play without jerking when playing from a CD-ROM. Converting to Movie jr. allows a user to recompress the movie so that it is optimized for use in a new situation.

It takes advantage of all the features of QuickTime 1.5 or higher and allows old movies to be converted to better compress routines, also known as *codecs*. For the first time, a QuickTime Movie can be updated to use the better technology in new versions of QuickTime from Apple. This is a cooperative effort of several of the DTS engineers that were at MacHack. The controls (see Figure 5.3) are well laid out and easy to use.

CountPatches by Brian Bechtel

7 §

There are bragging rights that are assigned by the number of patches a programmer does not have to make to get a program running. These contests begin to sound like the old game show "Name That Tune," with programmers telling other programmers that they can do it in 4; no, 3; no, 2; no, 1 patch. So, in order to validate all these claims, Brian Bechtel wrote CountPatches. It needs to load before any of the other extensions load. Once it is loaded, it indicates the number of patches an extension makes during startup. Brian, however, would not say how many patches his hack used and offered the code for programmers to use to make a manual count.

DesktopSwitch by Sam Madden

7 §

When people share a machine, they each have different ideas on how to arrange the desktop. Even a single person on a machine may want different files available on the desktop, depending on what they are doing. DesktopSwitch allows a user to create sets of icons that

should appear on the desktop and then allows them to load these sets, as needed. This allows a user to quickly switch between sets of icons on the desktop.

DiscoMac by Steve Bollinger

7

On Saturday Night Fever, the lights pulsed to the beat of the music; with DiscoMac, the screens can pulse to the beat of the music or sound that the Macintosh is playing. It is an extension.

Remember to remove it, or every time the Mac beeps, it will pulse the screen.

DontShareIt Lite by Jim Luther

7 § HW-CDROM drive

With an audio CD in a CD-ROM drive that is set up for file sharing, the disk becomes locked in the drive, if someone attempts to mount in on their machine. This can be really annoying when 50 or more people can be logged into a machine from all over a building. Jim Luther had this problem at DTS. His solution is DontShareIt Lite, an extension that checks to see if the CD is an Audio CD, not a data CD. If there is an audio CD in the drive, DontShareIt will not allow the volume to be seen over the network. If it is a data CD, then the CD can be mounted as usual. Jim included source in hopes that all of the CD-ROM vendors would fix their drivers to deal with this problem.

Drop Rob•Box—Icons by Rob Gibson

7

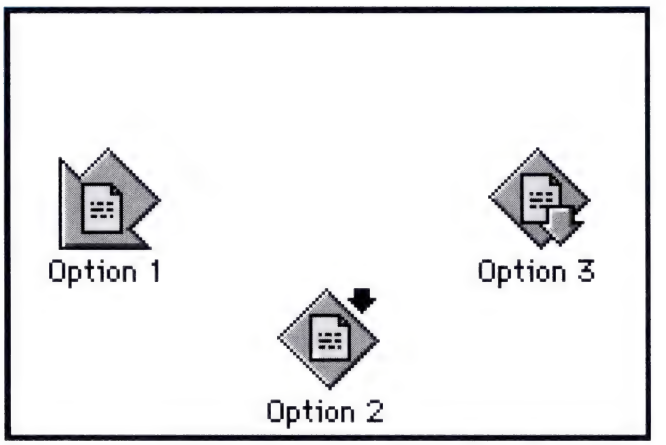


FIGURE 5.4 DRAG-AND-DROP ICONS

Rob Gibson is proposing that the current drag-and-drop icons are not unique enough. These three icons (Figure 5.4) are an attempt to try out new styles of drag-and-drop (droplet) icons.

Folder of Horror by Brigham Stevens

67

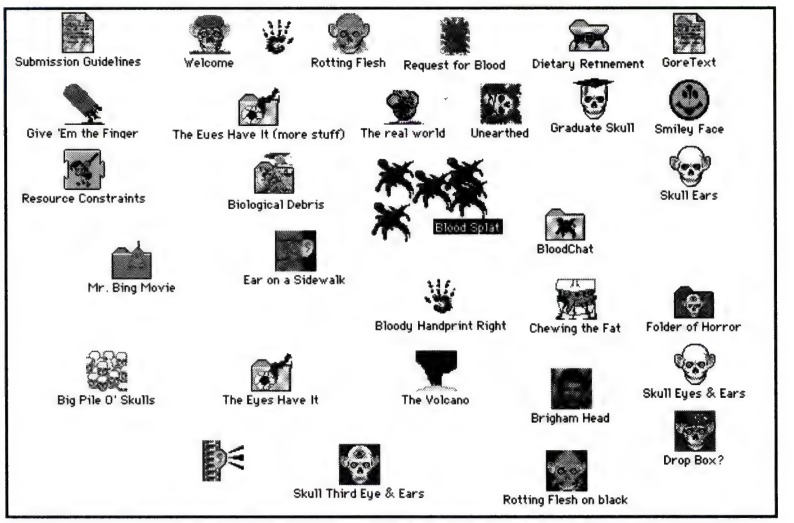


FIGURE 5.5 THE FOLDER OF HORROR ICON GALLERY

Brigham is a gifted artist and an avid game player. Many of the 1993 hack icons came from Brigham's palette. Trying for a new motif on the Macintosh (Figure 5.5), Brigham created the Folder of Horror. This rogue's gallery of icons will cause most people to leave your Macintosh alone, especially if they are used to working with MS Windows.

Hellcats FlightRecorder by Brigham Stevens

7 § SW-HellCats SW-QuickTime

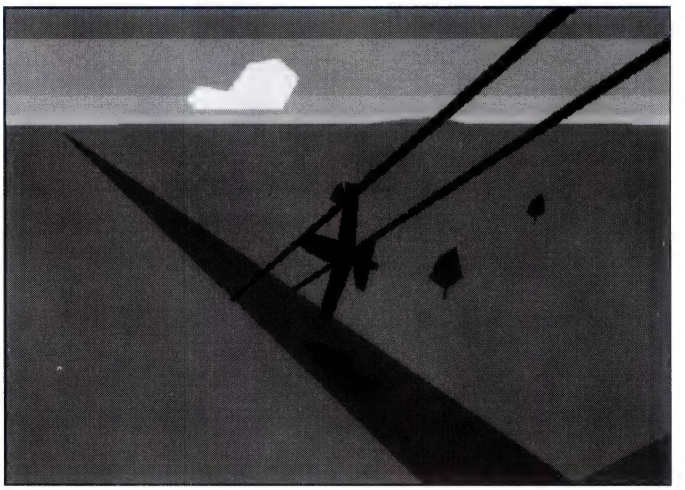


FIGURE 5.6 UNDER THE BRIDGE

Brigham (the master of horror) is also a gifted programmer. Hellcats is a great game to play, but explaining to friends that you just flew a rad mission can get you looks of disbelief. Brigham, who has flown several rad missions, decided to prove to his friends just how good he is. He created Hellcats FlightRecorder, a QuickTime-based screen recorder, which allows Brigham to refly any mission for his friends, anytime.

IconDisposer by Cameron Esfahani

7 §

Quickly opening and closing large folders is difficult where there are large numbers of custom icons in the folder. The Finder slows

down to get all the icons and to display them correctly. Cameron wanted to be able to find items quickly in folders that had lots of things in them. His solution was to make the custom icons "disappear." By telling the Finder not to wait for custom icons, IconDisposer makes searching a volume for an item much quicker. To active Icon Disposer hold down the **Control** key while opening the folder. The custom icons will still display without holding down the control key if the folder is opened.

Jasik Park by Several Tired Hackers

67

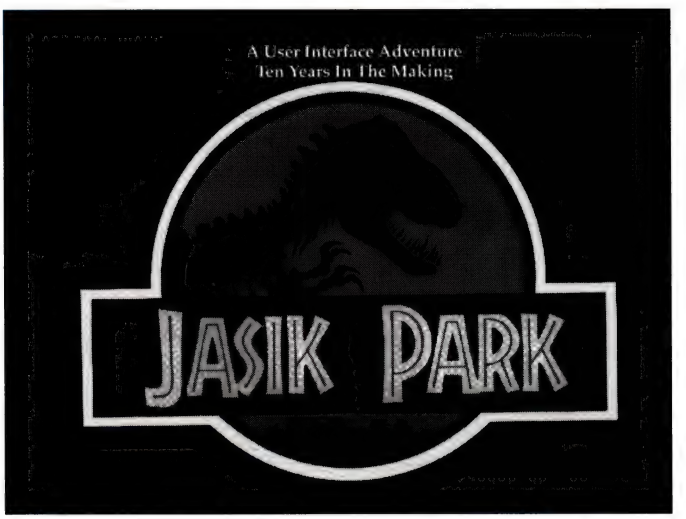


FIGURE 5.7 THE JASIK PARK POSTER

This is a parody of the Jurassic Park poster (Figure 5.7), which was created in honor (?) of Steve Jasik, the developer of MacNosy and other programming tools.

Jon's FKEYs by Jon Wind

7 §

Four function keys (FKeys) from Jon Wind, one of the top shareware tools creators, are listed as follows:

- *ClickLoc* will freeze the screen and give the coordinates of the box that the mouse can draw on the screen. This key makes it very easy to find sizes on objects.
- *FKeyList* provides a window with a list of all the FKeys that are installed on the machine.
- *FreeRAM* displays the amount of RAM (memory) on the Mac that is not in use in the menu bar.
- *GrabColor* will freeze the screen, and then when the mouse is clicked anywhere on the screen the color of the location is displayed in the menu bar in any of the normal color models.

JurassIcon Park by Mark Simmons

6 7

FIGURE 5.8 DINOSAUR ICONS

A post-hack show hack. Mark created six dinosaur icons that are very realistic (see Figure 5.8). When Jurassic Park was just released to the theaters, dinosaurs were on many people's minds at MacHack.

Listen to your hack... beat by Bernie Bernstein

7 § HW-MIDI interface SW-Macsbug

This is the software that Bernie wrote to go with his paper. MacsBug (Apple's Debugger) has to be installed, and Bernie's software has to be in the Macsbug DCMD folder (Macsbug installs it when the debugger is installed) in the system folder. Once all the pieces are in place, Bernie's code creates a MIDI file from the software that is executing. This file is then played through a MIDI interface. Microsoft Word creates some very interesting "Industrial" music when it is running.

Mom Hack by Jeff Walker

7

Any game writer should be quite aware of Jeff Walker. He likes to play games, but he loves to break into games and find the key points in those games where all the decisions are made. He writes code that will make it possible to beat the game [this is called InterApplication Breaking and Entering (IABE)]. Jeff normally calls the functions in the game directly and manipulates the results to his advantage. Jeff wrote Mom Hack which demonstrates the power of IABE. This was the first intensely well-received hack that was shown at the contest.

Mystery Science Mac by Jörg Brown, Bill Britton, and Jon Arkley

7

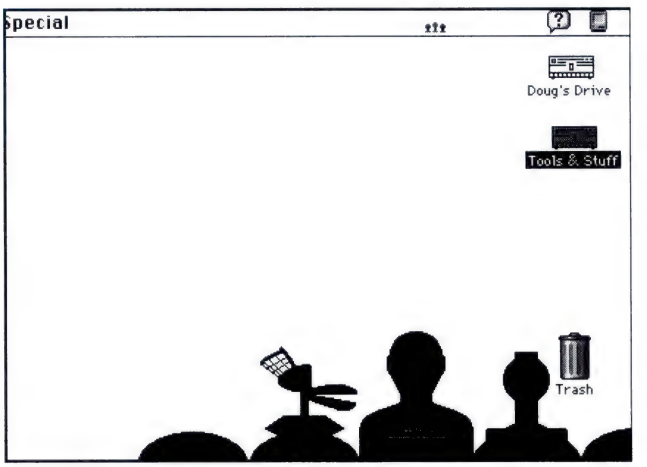


FIGURE 5.9 THE BOYS OF MST3K COME FOR A VISIT

Two little robots and a person appear on the bottom of the screen during really bad old science fiction movies on Mystery Science Theater 3000 (MST3k). Jörg, Bill, and Jon decided to invite the boys of MST3k to visit the Macintosh (see Figure 5.9). If Speech Manager and MacinTalk II are installed on the same machine as MSM, then the boys will talk while you work, just like they talk during the movies.

Neutron Bomb by Jeff Walker and Jon Wätte

6 7 SW-Spectre

Spectre is a tough game to get great scores on. The programmers at MacHack were trying to beat Jeff's and Jon's scores all week; but no one could. At the Hack contest, Jon explained why when he rolled out the Neutron Bomb. By pressing the **9** key, Jon froze the world

around him, as he continued to operate his tank. This brought hisses from some people and cheers from others.

OK, What was that again by Sean Parent, Scott Boyd, Eric Traut, Wayne Correia, and Nick Kledzik

7 §

OK,OK,OK, just pressed the OK button when it was installed. OK, What was that again, records the dialog before the OK button is pressed, so that the dialogs can be reviewed to find out why the print job is not sitting in the output tray of the printer.

PageTool by Brian Topping

7 § SW-PagerPro SW-MPW

Macintosh Programmers Workshop is not known for the speed of compiling new programs. Some compile jobs take hours, and sitting around and waiting for the computer to finish can be very boring. Brian Topping eliminated the need to wait for the compile to finish when he created PageTool. PageTool sends a message to a PagerPro server on the network, which in turn dials the phone and sends a page to the programmer. This means that the programmer can go ride the roller coasters, or shop for groceries, while the Mac is busy building great new software.

Parrot by Bernard Bernstein

7 §

Parrot will randomly replay bits of sound that it picks up in the microphone, at a random delay. It can make for a really confusing

party, if the microphone and the speakers are good enough. Great party hack by Bernie.

pLayer by Don Brown

7 §

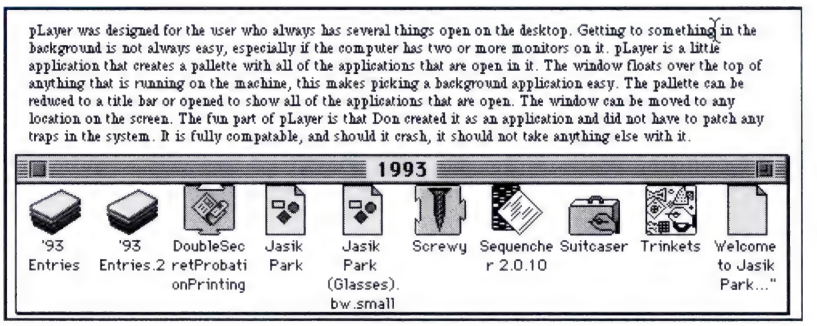


FIGURE 5.10 THE OPEN PLAYER WINDOW

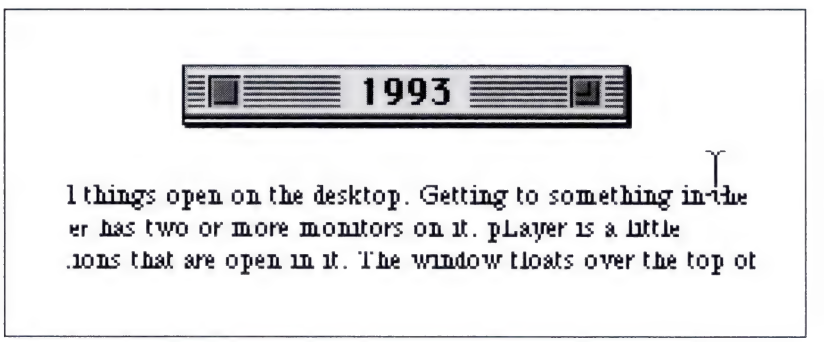


FIGURE 5.11 THE MINIMIZED PLAYER WINDOW

The second of three hacks that Don Brown of CE software created in an afternoon at MacHack, pLayer allows a user to select a folder and display its contents in a window that floats over all of the open applications. Clicking on any icon in the pLayer window will launch that item. pLayer's window can be either open (see Figure 5.10) or minimized (see Figure 5.11); this is done by clicking on the Grow box. Like AppBar, Don Brown, created this application without using any patches. It is an application, not an extension, and it has the same problem that AppBar does, in that it cannot be forced to quit; the machine has to be restarted.

QuickDat by Richard Zulch and Stephan Somogyi

7 SW-Retrospect 2.0 or higher SW-QuickTime 1.5 or higher

Quarter screen QuickTime has a low enough data rate that a Digital Audio Tape (DAT) drive can keep up with it. Richard Zulch, the creator of Retrospect, and Stephan Somogyi of *MacUser* decided to make use of this fact and the large storage capacity of DAT drives (2 gigabytes plus) to record movies. Using Retrospect as an engine to operate the DAT drive and an APS DAT Drive, Richard and Stephan were successful in getting QuickTime movies stored on the tape and in getting them to play back at 30 frames a second (full speed) from the tape.

Screwy by Francis Stanbach

7

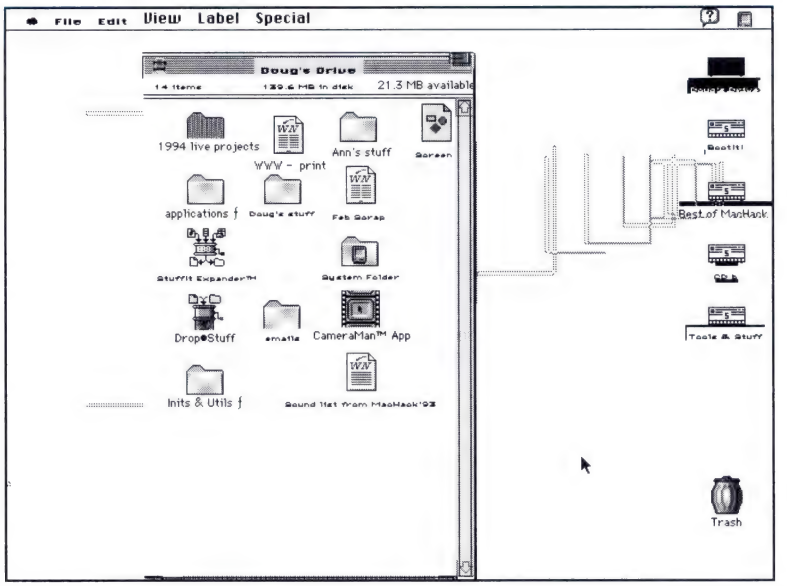


FIGURE 5.12 SCREWY IN ACTION

This system extension is the ultimate practical joke on the Macintosh. Install it and it will mess up the whole interface (see Figure 5.12). It is just over 2k in size, but it patches almost everything and the results are crazy. To remove Screwy, restart and hold down the **Shift** key to boot with extensions off.

WARNING

No real work can be done on a machine that screwy is active on.

SCSIPatch by Paul Baxter

7 §

SCSIPatch is a low-level tool that intercepts. It calls to the SCSI bus from the computer and the replies from the disks and other devices on the bus. The log file from SCSIPatch is very useful in writing SCSI drivers. It is also useful in finding out whether or not the drive and the Mac are well matched for each other. Performance was not an issue for disks and Macintoshes until full-screen video became important. Now it is critical. This tool, when used with a copy of the SCSI standards from ANSI, will provide the information required to tell if the drive and the Mac are both working in top form and if they are well matched.

SegaMac by Alex Rosenberg

HW-Sega Genesis

At the end of the Hack show, at about 2:30 AM, the DTS engineers were scrambling to hook up the hardware for the final hack. Getting that hardware of a SegaGenesis to hook up to a Macintosh projector was a great hack in and of itself. Anyway, Alex told the audience that he had been busy porting the Macintosh operating system to the Sega machine. He then proceeded to warn everyone that the hack was unstable, just finished, and would probably bomb. He started the Sega and the smiley face Mac showed up. He showed a couple of extensions loading and then the machine dropped to the debugger with Sonic the Hedgehog on the screen tapping his foot and waiting for the programmer to fix the problem. This hack, unfortunately, is not included on the disk.

Shaman by Robert Hess

7

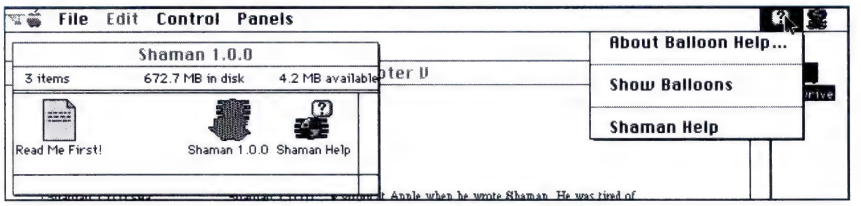


FIGURE 5.13 THE SHAMAN INTERFACE

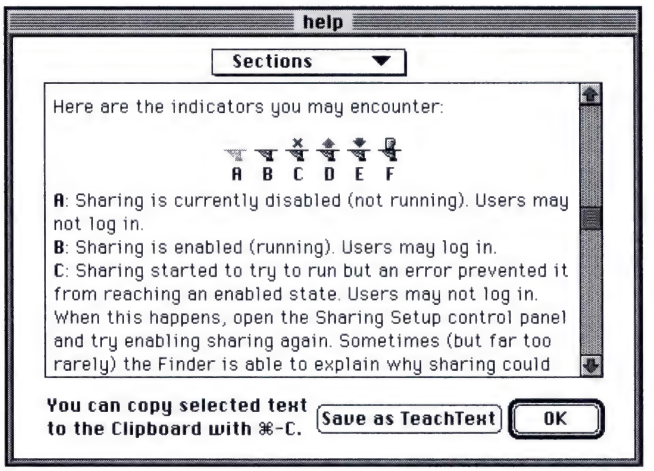


FIGURE 5.14 THE SHAMAN HELP SYSTEM

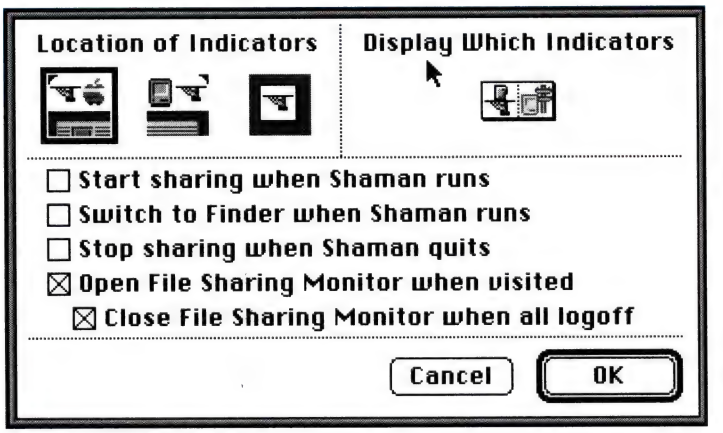


FIGURE 5.15 THE SHAMAN PREFERENCES

Robert Hess was a software engineer in the networking group at Apple when he wrote Shaman. He was tired of people wanting to know if ARA was on or off their machine. He was also tired of people slowing down his machine, because he had forgotten and left file sharing on. Shaman was written to give visual cues about the status of both these items (see Figure 5.13) and to also give the owner of the machine a quick way to change the status of either item. Robert even had time to create an extensive help system (see Figure 5.14) and patch it into the Balloon Help menu (see Figure 5.15). For a tour of shaman, launch it, and then check out the Shaman Help item that will appear in the Balloon Help menu. This application will run without ARA being installed on the machine, and it does offer full sharing control.

Sibling Rivalry by Matt Slot

7

Normally, an AppleShare server has several volumes (disks) on it. Going to the chooser to get the first one is not too bad, but continually going back to get additional volumes can be very time consuming. Matt's extension makes it a snap; just hold down the **Command** key and click on the AppleShare volume that is already mounted from that machine. A menu of the other volumes will appear; choose the new volume to mount from the menu and it will.

Smooth Updates by Mark Smith

7 §

On windows with many files and icons, the Mac can end up displaying first the frame of the window, then some of the items in the window, and then more items, and so on. This "jerky" display technique can cause a window to be closed by a user, before all the information is displayed. Support calls have been made because files seemed to disappear. Mark intercepts the drawing commands for windows and makes sure the window is complete in every way before displaying the window. He also makes the window updating a bit faster because of the way the windows are now handled.

Sort by Paul Baxter

7 §

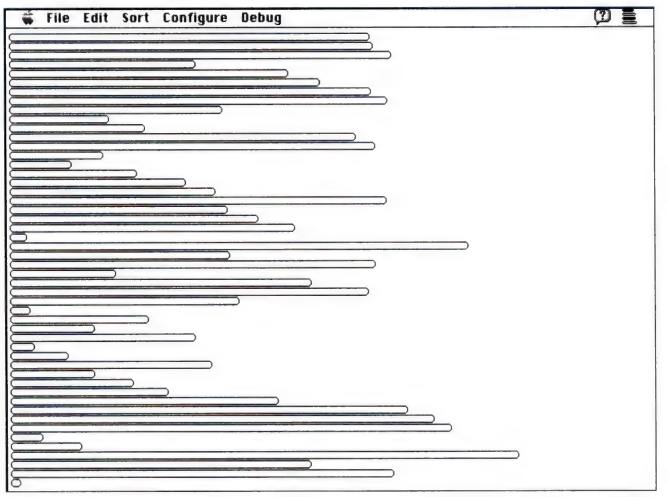


FIGURE 5.16 THE SORT INTERFACE

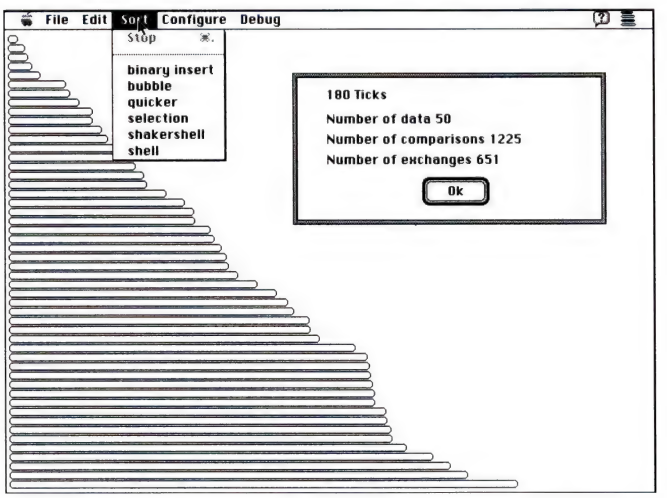


FIGURE 5.17 SORT OPTIONS

In every computer science program, there is a requirement to write a program that will sort data. Paul Baxter wrote a program using all the classic sorting routines (see Figure 5.16) and displayed the progress visually. This program is lots of fun to watch as it moves the bars up and down (see Figure 5.17). The program itself is not real interesting, but the source code for all the sort routines is included and the source is very well commented, making that computer science program a snap.

Suitcaser by Troy Gaul

7

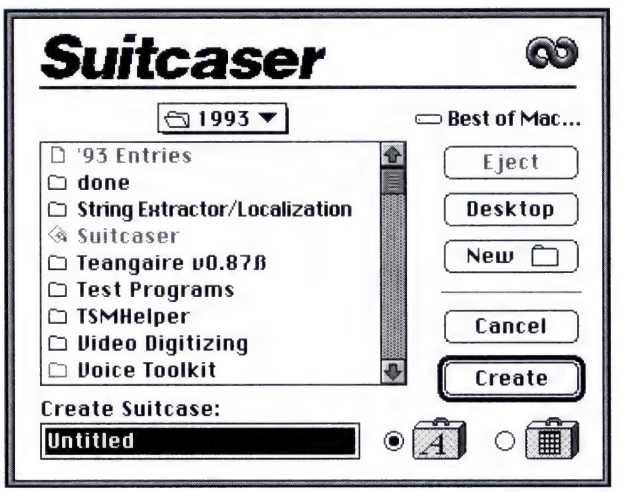


FIGURE 5.18 MAKING SUITCASES

When moving fonts in and out of the system folder, having an empty font suitcase can be very handy. The Finder cannot create one. Suitcases deals with this problem by creating either TrueType or Bitmap font suitcases that are empty (see Figure 5.18). They can be

double-clicked just like a normal suitcase, and the fonts can be pulled into them. This eliminates the need for having ResEdit around to move fonts or the old Font/DA mover.

Talking Compiler by Stan Shebs

7 § SW-MPW

Having heard Bernie's talk about code that makes noise as it executes, Stan Shebs built a compiler that talks while it is compiling. The compiler announces the problems with the code that it is compiling and also its location in the compile job as it goes. Jeff Holcomb created the sounds for the demo at MacHack. Because many of the sounds are from commercial sources, they are not included. Needless to say, the first time the choir sang "You are an idiot" the whole room exploded in laughter. The "read me" file includes information on how to customize the Talking Compiler for your own use. The full compiler is included.

Teangraire by Gary Kacmarcik

7 HW-68040 based Macintosh

Teangraire was the best hack pulled on a hacker. Gary had created a PowerPC emulator to run on the 68040-based Macintoshes to test the efficiency of code that was written for the PowerPC. Scott and Jordan learned of Teangraire and decided to play a little joke. They substituted the PowerPC from the locked room for the Quadra that Gary thought that he was running on. He then emulated a PowerPC on the Emulated Macintosh on a PowerPC. When Scott revealed this to Gary and the audience the whole room shook with laughter. It was a good test of the stability of the PowerPC emulator—both Apple's and Gary's.

Trinkets by Francis Stanbach

7

FIGURE 5.19 TRINKETS WINDOWS

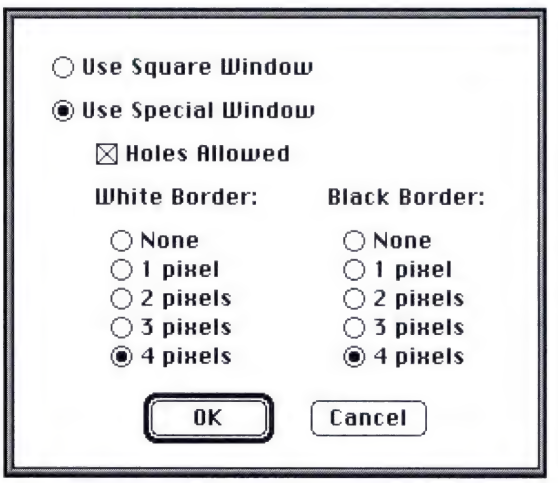


FIGURE 5.20 TRINKETS WINDOW CHOICES AND PREFERENCES

Francis stepped up and started his application after installing a new scrapbook. Everyone wondered, why a new scrapbook. Francis showed why when he pulled a picture of a banana from the scrapbook and pasted it into trinkets. A banana-shaped window appeared (see Figure 5.19). Next, he did an apple and then a fish. Silly window shapes are the neat thing about trinkets. More importantly, is that Trinkets could be used to make post-it notes, which can reside on the desktop to tell people where the owner of the machine is or for keeping to-do lists, or just for leaving a folder of fruit salad around. Trinkets is a fun program to play with. Trinket windows come complete with borders and holes in them, where there is no information, like the center of the letter O (see Figure 5.20).

TwilightZone by Steve Bollinger

6 7

There are hundreds of After Dark modules and no good way to check how cool they are without installing them in After Dark and restarting. This is a slow and painful way to set up a really cool screen saver using multimodule technology. Steve solved this problem by creating an application that will run After Dark modules in a window. This window will not be full size, but it is big enough to determine what the module looks like when it is running. This prevents having to install modules to see what they look like and it also avoids lots of rebooting.

Unca Fenster by Timothy Knox and Ed Tecot

7 §

Arranging a number of windows from several applications on the screen in order to be able to work on a complex project can mean several minutes will be spent moving the windows around repeatedly each time the project is started. Most application programs are not

smart enough to remember where the window goes and how big it should be, but Unca Fenster remembers! This application by Tim and Ed can be run to remember where the windows belong, so that if the tax returns need updating, the status of all the windows that were open during the last tax session can be returned to the correct locations. Running Unca Fenster for each project and renaming the data files can make it easy to share a Macintosh with two or three people.

Friday Early AM

The Hack contest ended at 3 AM, several people left the hack contest to return to the machine room. Very few people went to bed. A group of programmers walked about a mile to the ice cream store and found it closed, so they called a cab and went to an all-night grocery store and bought several gallons of ice cream, which they brought back to the hotel. They then realized that they forgot spoons and bowls, which lead to a very messy ice cream eating contest. Morning came very early for most of the hackers.

The main program did not pick up until 1:00 the following afternoon. Rick Fleischman covered Dynamic Linked Libraries (DLLs) on the Macintosh. This session looked at the component system that will become the software model for the operating system after System 7.5. This will allow programmers to directly hook into parts of the operating system and control them from applications. Additionally, applications can be built from a set of DDLs. These applications can then be modified by the users on their Macs by replacing one or more of the DLLs with more powerful DDLs from other vendors. This is the model that Microsoft uses with Visual Basic.

David Newman followed with a discussion of the Pen-based operating system for the Macintosh. This system was slated for use on some of the Duos that will ship in 1994. David showed how the Macintosh could be controlled with a pen rather than a keyboard and a mouse.

Waldemar Horwat followed with his annual paper, which dealt with parallel processing computers and how the chips in the computer talk to each other. Waldemar again proved that he is the master of the

arcane. His paper offered a look at the replacements for the mainframes and supercomputers that currently do the big jobs for corporations.

The paper presented and compared several approaches of viewing communication, including cache-coherent shared memory, message passing, and higher-level protocols. The important hardware constraints were examined, and some predictions for the future were presented. The audience did not fully appreciate what Waldemar was presenting, no one had gotten enough sleep. Weeks later, many people started commenting on what Waldemar had said at MacHack, after they had a chance to reread the paper and understand the impact of this technology.

Jim Laskey from ProGraph finished the afternoon by offering a look at the state of visual programming. Jim's talk ended just in time to catch the bus to the showing of Jurassic Park, which the theater was doing just for the MacHack members. They had tried in previous years to mix the regular audience with the MacHack crew, but they had too many complaints from the people about the comments that were being made during the movie. After the movie, we visited the ice cream store (now open), and then most of the programmers returned to the machine room to do a bit of programming. Several new hacks were created that night, as people were inspired by what they had witnessed the night before. One hack in particular is of benefit to Apple and the antiviral software writers. Several of the system software engineers and John Norstad (Disinfectant) wrote a program that exploited a weakness in the operating system. Because they had a working model of the problem, Disinfectant and other antiviral programs now look for this kind of virus and stop it from ever working. Apple used the information to change the way that System 7.5 works. This change will prevent this kind of virus from working in the future.

Because of the hack's nature, it is not included on the CD. It has become known in the programming circles as the "evil disk hack." Because of misinformation on who wrote it and the distribution or lack of distribution of the hack, several angry letters were exchanged between members of MacHack and institutions of higher learning. The hack has only been given to people who need to understand it to build defenses against it. No distribution of the hack beyond that was ever planned. This is an example of programmers trying to prevent a problem, and because of misinformation and rumors, it becomes

"Danger—hackers on the loose." Most of the fallout from Evil Disk has been cleaned up. There are a few people who are still sure that the hack was done to destroy their machines and to make the Macintosh unusable. Given the group of very senior people who tackled this problem and the refusal to discuss even the outline of what it does or how it works, most people are now sure that it was done for a good cause. Only time will tell how the evil disk story plays out.

On Saturday, a small group of programmers attended the local user group meeting and reprised the Hack show. Most of the hacks were not shown, only the winners. The local user group normally draws an extra hundred or more people to the meeting when the MacHack hacks are going to be shown off. MacTechnics, the user group, later bought pizza and soda from the programmers.

During MacHack, several people collected quotes, which are listed below as they were uploaded to the networks after MacHack. Several are inside jokes, but most of them stand on their own:

"My pad of paper is not backlit." —*Greg King*

"I like to do very very late binding. 'Just in time' binding." —*Dave Feldt*

"Windows doesn't use AppleEvents."

"TextEdit does everything right." —*Jon Wätte* (Sorry I couldn't spell your name correctly, but I couldn't find the key.)

"Using link is noise." —*Alex Rosenberg*

"If you want to see a pipelined execution model, go to a laundromat." —*Richard Clark*

"It's a SCSI problem." —*Ed Tecot, on why dinner was slow*

"Smart Friends ask no SCSI questions!" —*Apple employee at the Bash*

"Name them after trees." —*Suggestion on how to improve the identification of tech notes*

"No one up here is unsympathetic with your desires to have tools as good as you remember." —*Apple employee at the Bash*

Here is a new koan: "Will there ever be a C++ version of TCL? Bedrock." —*Mary Lynne Samford*

"How do you partially execute a program?" —*Dave Neal*

"No, it's not me, it's the glue." —*Paul Cambell*

"When I started hacking, we didn't have ones. We had to make do with zeros." —*Overheard by Andrew Craze*

"[MacHack] is your brain on Jolt." —*Bill Fernandez*

"How did you fix the stack without BlockMove?"

"We have the Memory Manager source code, the Resource Manager source code, and the QuickDraw source code."

"It's not a bug. It's a warning." —*Richard Clark*

"Something horrible is wrong." —*Steve Falkenberg*

CHAPTER 6

Mozart, Gershwin, and Copland

Mozart, Gershwin, and Copland are the code names for the next three versions of the Macintosh Operating System. Mozart is also known as System 7.5 and will be released in the fall of 1994. Many of the hacks from previous MacHack contests have helped shape these new versions of system software. Many of the programmers who came to MacHack fresh from college were hired by Apple to create the future versions of the Macintosh.

MacHack '94

On Friday of MacHack 1993 there was a planning meeting for 1994. Everyone there agreed that the Macintosh was getting to be long in the tooth. Scott Boyd was the first person to speak. He told the group that the Macintosh was getting old and boring, that he wanted to move on, and that if MacHack wanted to stay fun, that they should consider moving to a more open model. Let General Magic and others offer incites on their operating systems at MacHack. Leonard Rosenthal and the others agreed. They wanted to see more platforms covered at the conference. Waldemar and some of the others were worried about the conference losing focus. The discussion lasted

for over an hour; it went back and forth over what MacHack should be and even touched on the future of the Mac. Finally, everyone agreed that the conference should expand its horizons. The programming chair was directed to keep a Macintosh focus, but to open the conference to other "interesting" platforms. Interesting was never defined and the program chair was on his own to make a decision on who to invite.

With that in mind, the group at the planning meeting spent an hour trying to decide who best represented the shift in the programming community. Finally, someone suggested Andy Hertzfeld and everyone agreed Andy would be perfect.

Darin Adler and Greg Marriott, both ex-Blue Meanies and long-time MacHack attendees, were at General Magic and willing to try to get Andy for the keynote speaker. General Magic has created the next generation of operating system. MacHack needs to move to the next level: new tools and new operating systems.

Andy was well known in the Mac community. Andy was on the original Macintosh team; he was a long-term Apple veteran considered by many to be the innovator for the Macintosh's Graphic User Interface. His split with Apple is legendary. Andy wanted to see the Finder change radically, as he felt that the Macintosh interface was dated and was not quite right the first time. He worked hard within Apple to get the changes he felt were important included in the newer versions of system software. Finally, he chose to go to the users of the Macintosh and let them decide. His option, his soapbox, was Servant.

Servant eliminated most of the scroll bars and controls. It did away with the Font/DA Mover and many other items. In its place was the hand cursor, which could move things, pull or push the window around, and open files. It also gave direct access to the system file. This allowed users to directly replace fonts, change sounds, and reword dialog boxes. Servant was a hit with the programming community, but it was not a hit at Apple where they were struggling to define the future of the Macintosh User Interface. Andy and his boss fought about Servant. Several Apple engineers denounced it on the nets. Rumors about Andy's code and how hard it was to debug ran around the pro-

gramming circles. It was obvious to the programming community that a struggle for control of Macintosh's future was taking place. Andy was getting good press and a good play on the nets. Many pointed to the fact that Servant had been in Beta testing for over a year and was not finished or fully stable. Andy and Servant were the subject of many discussions during the time all this happened.

Finally, Apple paid Andy a fairly large sum to give Apple the rights to Servant. Andy turned the code over to Apple, who in turn took a few of the ideas from Servant and included them in System 7. System 7 sports the ability to directly get to fonts and sounds without needing an application to load and unload them. The Desk Accessories, under the Apple Menu, are in their own folder, too. For most of this ease of use, Andy is the innovator that needs to be thanked.

Andy moved to General Magic, taking his payoff for Servant with him. General Magic's start-up was funded by Apple's venture capital arm. He worked with another long-term Apple person, Bill Atkinson. At General Magic, Bill and Andy were creating a new operating system called MagicCap. MagicCap is the first operating system designed from the ground up as an OS for communicating between computers. The key to MagicCap is the ability to send small programs to other computers, so that these programs, or agents as they are called, can then carry out a specific task. For instance, an agent could go to a airline computer, select a flight, get a seat assignment, and pay for the transaction. All of this activity would be programmed in a building block manner by the owner of the machine. The software would be smart enough to remember things like window seats are preferred. This kind of an operating system in a small hand-held device should allow travelers and busy executives to make connections not only with the airlines, but with their offices and with their customers. The operating system borrows many elements from HyperCard, Bill's last major creation at Apple.

It also borrows bit and pieces from the Servant application that Andy created. Both parts of the heritage are visible in the final product. Andy said, "yes," only a few weeks after MacHack '93, which gives MacHack '94 a powerful anchor.

Surprisingly, after being spurned by Bill Gates in 1993, Microsoft is one of the major sponsors of the 1994 MacHack. Microsoft's tools group is making a major push to get Macintosh developers to both port programs to MS Windows and to develop Macintosh applications using Microsoft's forthcoming Macintosh Visual C++ development kit. The development kit requires that a Windows NT machine be available to be the actual development platform. Add a Windows machine, and a Macintosh and development of Macintosh native applications is now possible.

Additionally, there are new players in the Macintosh tools market with Metrowerks and Novell each offering significant new tools. Both of these companies came on board as sponsors and agreed to send speakers. Add to the mix 3DO and Taligent with new operating systems, and MacHack '94 is looking very full.

Because of the many people who had to stay over Saturday night, the committee moved MacHack from a Wednesday to Friday conference, to a Thursday through Saturday conference. This opened an important opportunity: the ability to have tutorial classes during the Monday through Wednesday time period. So before the first session of MacHack 1994, Metrowerks and Apple will each present a three-day tutorial during that three-day period in 1994, and in 1995 it looks like there will be as many as four tutorials going during that three-day period. The shift in days also allows the conference to hold the banquet as a closing event rather than an opening event. This means that the Hack awards will be presented at a semiformal banquet. (How formal can 300 programmers with nerf weapons be?) The event will allow MacHack to showcase in a single event the best software that is created at MacHack. Apple waking up to the power of the MacHack Hack contest has implemented a similar contest, thought it will not be judged by the attendees and it will not have a midnight hack show. The mystique of the MacHack show will remain. The influence of the 1994 Hack contest will be felt on several platforms, not just the Macintosh.

MacHack '94 is making maximum use of the three days that the conference is scheduled for, the keynote address will be delivered Thursday at 12:01 AM. The hack show will be Saturday at 12:01 AM and the movie will be Sunday at 12:01 AM. Additionally, sessions will

run until 4:00 each morning and start again at 9 AM. This will allow MacHack to present over 80 technical sessions and 10 papers in just three days, with no more than two sessions running at the same time. Most of the new technology from Apple, Novell, and General Magic will be presented. Microsoft, Symantec, Metrowerks, and Prograph will all provide an in-depth look at the tools that are available from them. In short, MacHack will cover many of the third-party areas that WWDC seems to miss.

MacHack will start with Andy Hertzfeld's keynote at midnight. He is expected to cover both the history of the Macintosh and the future of both the Macintosh and of General Magic's MagicCap software. Following Andy will be the traditional "Welcome to MacHack" session with Carol Lynn and Marshall Clow. Marshall is the chair of MacHack for 1994. He found that when he left H-P and joined Aladdin Systems that there was time available in his life for both his family and MacHack. Following Marshall and Carol will be Greg Marriott with a session on the rules of the Hack contest. He will also cover topics such as how to hack and what kinds of hacks have won before. The legends of hacking will be reprised for all of the newcomers to MacHack. Once Greg finishes at 3 AM or so, sessions will stop for the evening. Programming in the machine room and elsewhere will continue. The lights in the public areas are never turned off during MacHack, and the Holidome is seldom empty.

Microsoft will lead off in the morning with James Plamondon giving a talk on "Windows for Mac Programmers." He will discuss the topic in-depth in a special 4-hour session that will contain information on the upcoming versions of Windows and Windows NT, so that Mac programmers will have a chance to be in step with the future on those platforms too.

Following James will be Spriteworld, presented by its author Tony Miles. Tony was an Apple engineer and a gamer, who felt that the lack of sprites on the Macintosh hindered good game development, so he wrote what many have called the Sprite Manager. Tony has proven that sprite-based Apple II and MS-DOS games can be moved quickly to the Macintosh when Spriteworld is used as a library in the program. Spriteworld offers many developers of games a significant savings in time and energy when they are developing games. No longer does the

Macintosh code have to be completely different from all the other platforms. The second paper of the morning will be "Implementing a Zoom Control" presented by Kevin Killion. Kevin will look at the different ways that software has implemented zoom control on the Mac in the past and will offer options for future implementation. He also has what may become the de-facto way of implementing zoom controls. It is more intuitive than most of the previous efforts.

David Winer, the president and founder of UserLand software, will follow with a discussion of Frontier as a scripting language for use on the Macintosh. He will be presenting ways that programmers can build more power into their applications without having to do any more real programming. John Powers of Apple has a paper on the implementation of "Apple Help." John will cover some of the unconventional ways to use Apple's new help system. Stephen Humphrey will then present a paper on "Developing Cross-Platform OpenDoc Editors." The editors are the heart of the OpenDoc system. Stephen's paper will be loaded with the practical experience that he has gained while developing WordPerfect's OpenDoc software.

System Object Model (SOM) is a topic that will affect everyone in the future. John Arkley, an Apple engineer, will be presenting SOM at MacHack. It defines how component software should be built and how the modules should talk to each other. This model was developed by IBM and is endorsed by Apple and a large number of other vendors. OpenDoc will be built using the SOM standard, as will most of the future Apple system software. This means that in time, third-party finders may be available for the Macintosh, making it possible to customize the interface for a specific task.

Following the SOM session Dylan will be presented by Mike Lockwood. Dylan was to be the base language and the basis of the operating system for the Newton. It was not finished in time, so the Newton team took a different route. Dylan is a very high level language, it offers the power of hundreds of code lines in a single statement. Dylan may be the language that will link all the components of OpenDoc in the future. Dylan also offers promise as a way to unify the various application scripting languages that are in use today. It is similar to a very powerful object basic.

The afternoon is devoted to the first two of six sessions on the Newton. This track of programming should allow programmers to gain enough knowledge to create Newton hacks during the conference. ApplePIE is providing Newtons for people to try during MacHack and also to learn to program on. The first ever Newton Hack awards will be given at MacHack 1994. Alex Rosenberg will offer a complete look at what has changed in System 7.5, also known as Mozart. This release of system software will be the first to include Apple Help, TCP/IP the UNIX networking software, and many other features. It will reduce the number of items that are extra cost system software options. It is the first real update of System 7 since its 1992 release. System 7.5 is scheduled for release in the fall of 1994. Plaintalk, Apple's voice software, will be dealt with by its engineering team head by Tim Schaaff. This session will get into the details of making PlainTalk work inside of an application. Some of the details of this session will be discussed for the first time outside of an Apple facility.

Apple's version of the PowerPC allows programmers to create programs that are a mix of old Macintosh code and new PowerPC code. This allows programmers to quickly move a program to the PowerPC and only change the code that really makes a difference for performance. Mixed Mode, software that contains some native PowerPC code and some Macintosh code from the older 68,000 based machines will be presented by Eric Traut, an engineer at Apple who is responsible for supporting the development of mixed mode applications by third-party developers. This session will concentrate on the do's and the don'ts of creating mixed mode applications.

Apple is in the process of updating the CommToolBox, the set of tools that allows the Macintosh to talk to a modem or another computer. CommToolBox will be replaced by the Open Transport Interface (OTI), which provides a foundation for the future. It is based on the capabilities of the GeoPort, that Apple will soon replace the familiar serial ports with. The GeoPort will support full-speed transfers on modems faster than 9600 BAUD. This will be the first time that the ports on the machine will be able to fully use the 14.4 and 28.8 modems. It also supports directly ISDN bridges and even T-1 leased lines. This means that the future Macintoshes will be ready for the digital superhighway that everyone is talking about.

The last session of the day will be on PowerPC debugging. This session will focus on how to debug PowerPC applications. The session will draw on the experience of David Shayer, the developer of Public Utilities and Disklock, to look at all the new wrinkles that a developer needs to know in order to debug programs written for the PowerPC.

After dinner, it will be time for the annual Bash Apple session. For the first time, there may be more people on the Apple side of the table. The session will once again focus on the problems that people are having with Apple and what they would like to see happen to fix those problems. The session will end with the time-honored Nerf fight. Sometime before midnight, a truce will be called in the Nerf fight and it will hold until the end of the banquet on Saturday. Once the hotel has cleaned up from the Nerf fight, then Steve Jasik, the developer of MacNosy and The Debugger, will begin a 4-hour marathon on debugging. The session should finish before the 4 AM cut-off for sessions, but if it runs true to form, it will not finish before the sun rises.

Eric Shapiro will be one of the first speakers on Friday morning; his session will cover Fat Patching. Fat Patching is the method that a programmer uses to create a single program that is both native to the PowerPC and is also able to run on a 68,000-based Macintosh. Eric has created versions of Video Beep and Spectator that take advantage of the Fat Patching methods. He has also taught Fat Patching for Apple's Developer University.

While Eric is warming up, James Plamondon will again be starting a 4-hour marathon. His topic will be "Writing Portable Applications with Visual C++ 2.0 for the Apple Macintosh." James will discuss how the Microsoft Visual C++ system will work with that Macintosh. Donald Olson follows Eric with his session on AppleScript. Donald is one of the engineers at Apple responsible for the development of AppleScript.

Steve Falkenberg of Apple's Developer Technical Support (DTS) group then follows Donald with his session on Porting to the Power PC. Steve will dig into the techniques required to move code from the 68,000-based Macintoshes to the new PowerPC machines. This promises to be one of the most popular sessions.

Dave Evans of Apple will follow with a session on the Drag Manager. The Drag Manager makes it easy to create drag-and-drop

applications. The newer versions of the Drag Manager offer even more features. David will finish the session by getting feedback from the attendees to take back to Apple.

John Arkley will begin the afternoon with a session on the Code Fragment Manager. This software is designed to make the housekeeping tasks in the RAM of the Macintosh easier for the programmer. Normally, programmers have to mark code in memory as purgeable, then go back and purge it. They also have to track the location of the code in memory.


NOTE

Most programmers can circumvent this problem by *locking code* in place in memory. Locking code means that the memory manager cannot move code around. This results in a fragmentation of the memory that is available and prevents full use of memory. The Code Fragment Manager handles these tasks and more for the programmer.

For many years, the Main Event Loop and memory management have been the two easiest ways to crash a program. With the Code Fragment Manager in place, the memory management becomes almost trivial.

Rick Fleischman follows with a session on the core tools that Apple will be producing for the future. This includes a look at the future of Bedrock, MPW, and MacApp. Additionally, there will be sessions on AppMaker II, OLE II, and Appware, which will be presented by the vendors of those products. In each case, an engineer will present and demonstrate the products, welcoming questions from attendees. The day will end on a lighter note when John Calhoun, the creator of several Macintosh games, and Tony Miles discuss the Macintosh and gaming.

During the day on Friday, two papers will be presented. The first is by Chris Russ dealing with Fast Fourier Transforms. This session will look at the key software for moving video and voice to the desktop. The look will be at the lowest level of the software, the algorithms that are the foundation of the code. Chris has a unique background that includes a large amount of work with coding algorithms for mathematics and scientific applications. He is now looking at other ways that math tricks can be applied to the software. Mark Smith from the

University of Michigan will present a paper on the max.500 standard, which is emerging as a new computing standard.

At midnight, the annual Hack show will begin. There are already over 50 hacks that have been discussed, most of them are partially complete. For the first time, there will be trophies for the best hacks, provided by Metrowerks. Additionally, Microsoft is providing pizza and drinks at half-time during the Hack show. Several senior designers for Microsoft, Apple, and others will be in attendance. Many of the system hacks that are shown here will end up part of Copland and in other new operating systems.

Saturday morning begins with a look at the General Magic software and its use. Darin Adler and Richard Clark will present three sessions on MagicCap during the day.

Rick Fleischman will examine in detail the Apple development frameworks. He will delve into MacApp and Bedrock to show how they can be used together. Donald Olson will then present a session on creating an AppleScript Editor in an application. He will use HyperCard 2.2 as a basis for this discussion. Kurt Piersol, the architect of OpenDoc, will offer four hours on the topic. Kurt is a speaker that can captivate an audience and transport them through time and space. Listening to Kurt is like stepping into a time warp. On one hand, it feels like no time at all is passing, yet the information content of the sessions tends to make the mind feel like it just had 40 hours of intense seminar poured into it.

Allan Gunn will follow Kurt with a session on the foundation classes of Appware. Appware from Novell is the "horse to beat" in the race for the best cross-platform object software framework. Appware, once called Serius, has grown and evolved under the care and feeding of Novell. This session promises to show the programmers not only the power of Appware, but the software's flexibility and the interface that will let programmers extend the foundation classes. It is better than Visual Basic with Visual C++. Additionally, there will be sessions on Hypercard and the SCSI Manager. Five more papers will be given that afternoon.

There will be an update on the Clipper chip and the encryption of data. The Clipper chip answers the government's need for data security on the networks. It is supposed to be almost unbreakable, but the government holds the key to the encryption system so that they can listen in anytime they deem necessary.

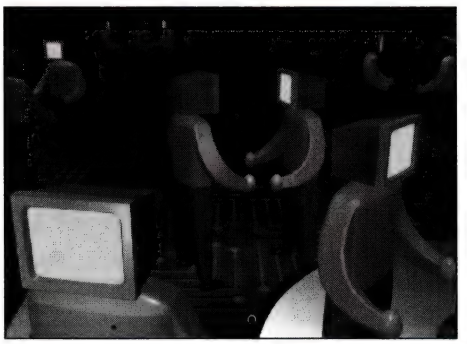
During all of these sessions, Leonard Rosenthal and his crew will be hosting Code Clinics for people who need them. This is the fifth year that Leonard has assisted programmers with problems in their code, not only in the code of the hacks that they are working on, but in the projects that they are actually producing. Finding the bugs in the code and how to solve them is Leonard's specialty.

In short, Saturday will also be full. The day will close with the hack awards banquet and the resumption of the Nerf fight. At midnight everyone will be declared killed in action and the hackers will adjourn to the local theater for a midnight movie. That will be followed by a trip to Baskin-Robbins for ice cream, the final event in MacHack '94.

MacHack '95

MacHack '95 is already taking some shape. Gershwin will be shipping by MacHack '95, if Apple keeps its schedule. The hacks from 1992 and 1993 that were incorporated into the operating system will see light in Gershwin. Taligent and Kaleida both have indicated that they will make presentations in 1995. Also, Microsoft is talking about a larger role. They are even talking about bringing several Windows NT machines to place in the machine room. Apple has indicated that they want a larger role in determining MacHack's future and have even reopened discussion regarding a possible MacHack Europe or a MacHack Asia. The committee for 1995 has not been chosen yet, but the underlying assumption is that if it's June and it's Ann Arbor, then it must be MacHack.

APPENDIX A



Hacks by Functional Area

Antiviral

Applescript™ Worm Folder
INIT 31+ ("Kills Dean's INITs")
Disinfectant

Demonstrations

Quorum Compatibility Engine

Future Operating System Enhancements

!MultiFinder
AppBar
DataStack Filer
Desktop Secretary
DragWindow
Drop Rob•Box—Icons
DropSave
Picker Placer
Finder Hack
Fred's Finder Hacks
Savvy
SCSIPatch
ShrinkToFit
Smooth Updates
StickyClick 2.0
Trash Selector
TrashCanNotifications

Games and Silliness

3D Bouncing Ball AD Module
AniMicons
Annoyance Pack
Barney
Busy Bo
DOS sHELL INIT

DiscoMac
Eric's AppleMenu Hack
Eric's ColorWheel Hack
Extended Warranty Mgr
Folder of Horror
Fuzzy Balls
Hellcats FlightRecorder
Insomnia
It's Just a Clock
Jasik Park
JurassIcon Park
Kilroy
LoonyHelp
Momentum
Moof
Mystery Science Mac
NetBunny
NetBunny 2½
NeVR!!
Not
not!
Neutron Bomb
Rapmaster
RISCy Bitsness
Run & Stumpy
Scott's Analog Clock
Screwy
Scribble
Too Many Lawyers...
Trinkets
TwilightZone
Unca Fenster
Ümlâût Õmêléttè
Windows 3.1
Wheel

Miscellaneous

Bail
BlueDot Hack
Conan the Librarian
FX
HyperInitMaker
Move Around!
MS Works Merge Enhancer
NextPrev
SpaceMaker
The Mac Clapper
The Regulator
The Regulator 1.4
ZoomOther

Multimedia

ColorHack 1991
ConvertToMovie Jr.
MacHack Weather
Mr. Bing
QuickDAT
QuickTime Balloon Help
Slack PICS
VolumeDock

Music

Bell Choir
DylanTalk

Listen to your hack... beat
MIDI DRVR
Parrot
Talking Compiler
Voice Toolkit

Networking

AppleShareSetup
AppletalkOffIIb2
ClipShare
CloakShare
DontShareIt Lite
MountAlias
NET/Mac
NetMouse
NetWarmer
Network Digital Video
Send the Hack
Server Remote Control
Shaman
Sibling Rivalry

Powerbooks

ARAStatus
battmonitor
Battery Indicator 140/170
PB Keyboard Remapping
Powerbook Compatibility INIT
PowerBook Pixels™ 1.0.1
Sound Asleep
Strobe

Printing

CoolLW
ChooserHack
DTPrinter

Programming Examples

Clarus—the Tail Patch
CountPatches
Mom Hack
Norstad Hack
Restart Test Hack
ReturnOpens
sort
StandardGetFiles
SuperGraphics
Surovell Stuff
SwapMenus
tclObjName
TCP Thermometer
The Essential Hack
Vector
Windoid

Programming Tools

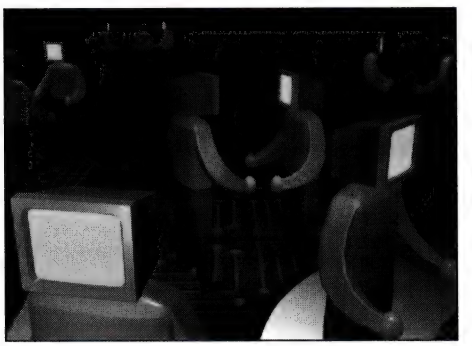
Æ RPC Stub Compiler
Apple Menu Munging
Bully
BNDL Hacks
Clicks, not!
INIT Maker
Localizer

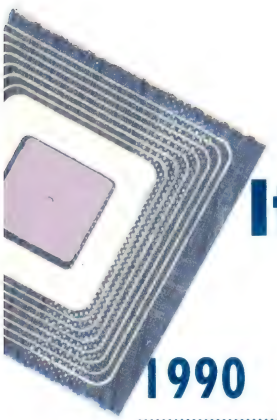
Speed Shifter
Think C Beeper
CommanderTabs
MacsBugTool
PageTool
Primes
Procedure Call Logger
ProcessFinder
ProcessWindowMenu
Rotate Process
String Extractor/Localization
Teangaire
TSMHelper

Utilities

AliasThis!
Ashtray
AshtrayII
Breakout Screen Saver
Change Type & Creator
Clock Menu
DesktopSwitch
DynaRamps
FinderKeys
FinderPict
FinderMenu
Grabber
Help for Hier
IconDisposer
iconEdit
IR Man™
Jon's FKEYs
Makin' Copies
MountImage
NotificationMon
NotificationTrash
open,open
OkOkOk
OK, What was that?
OK, What was that again?
PoliteNotification
pLayer
ReMounter
ScreenSaver?!!!
Scroll O Rama
ShowColor
ShowINIT Names
Silent Alarm, not!
SloppyCopy
SFComment
Suitcaser
Task-It
Text Capture FKEY
TrashMan
Zync

APPENDIX B





Items that are not Hacks

1990

Proceedings

A/Rose Techniques • Mark Rustad's presentation on A/Rose with sample source code. This folder includes Clister, an MPW tool for use with A/Rose.

Extensible Apps Paper in Word and MacWrite formats. *Building Extensible Applications Using Object-Oriented Methodologies*, by Shane D. Looker. This paper presents a way for application writers to allow extension modules written outside of the application development environment to be utilized by end users. This method allows users to select custom modules they desire in a product and create a "roll-your-own" application suited more directly to their needs.

Fortunes From Banquet, by Eric Shapiro. This is the complete list of fortunes from Eric's fortune cookies.

John Norstad's folder contains an early version of Disinfectant, the code for the user interface to disinfectant, and John's presentation on reusable code. The sample source code can be used to build other applications.

Power of C++, by Waldemar Horwat, is one of Waldemar's 1990 papers that deals with how C++ was used to build TMON Pro.

Semantics, by Waldemar Horwat, is Waldemar's other paper. This paper is an introduction to the field of semantics of programming languages, describing three kinds of semantics—operational, denotational, and axiomatic—and a most primitive programming language called the lambda calculus.

Session Schedule 6/14. The full schedule for MacHack 1990 in Microsoft Excel format.

TDynamicPopup is a paper presented at MacHack 1990 on Dynamic Popup Menus.

Attendee Contributions

In the '90 Other Stuff folder are utilities and other material that the attendees contributed at MacHack. The material is best used with System 6. Most of it has never been tested with System 7. Many of the items here have been updated by the authors; the latest versions are available on on-line services like America OnLine.

In the '90 Source Code folder is source code that the attendees contributed. This source code is meant to be used with System 6. Again, there are updated versions of the many items here available on the Internet or on other on-line services.

1991

Proceedings

Attribute-Based File Naming, by Michael McClennen, The University of Michigan. How much of the time spent using a Macintosh is devoted to searching for files, to organizing and reorganizing a hard disk, or to worrying about the placement of a file? The improvements to the file

system promise to simplify the process of organizing and finding files on a disk. But is this the best solution to the problem? It retains many of the limitations of the old file system and adds another layer of confusion to the interface. The key to this dilemma is the basic conception of the nature of a file name. With some thought, it is easy to interpret a file's path name as a set of attributes of the file being named. This viewpoint offers a way to restructure the file name space to overcome the limitations inherent in the current scheme, and open new possibilities for the organization of data.

Compilers, by Waldemar Horwat. Presently, almost all Macintosh software development is proceeding in compiled languages such as C++, Pascal, and even Lisp. The quality of the code generated by the compilers can have a large impact on the size and speed of the resulting applications and system software. In addition, poor compilers can have a detrimental impact on the readability of code as programmers try to hand-optimize their programs and discuss some of the key program optimization strategies. Although the current generation of Macintosh compilers produce adequate code, they lack some important optimizations. The applications of compilation techniques are not limited to compilers. Optimization techniques are now used in a wide variety of products, ranging from databases to debuggers.

Courseware Development, by Thomas E. Ludwig, Hope College. Courseware developers can choose from a variety of programming environments on the Macintosh, ranging from standard programming languages to special-purpose tools for creating instructional modules. This paper compares three representative development environments: ZBasic, a powerful BASIC compiler that offers total control of the Mac interface; HyperCard, the standard for courseware development; and Authorware Professional, the most powerful (and most expensive) of the instructional development systems. A sample courseware module illustrating the structure and function of the human auditory system (taken from the PsychSim courseware package), developed in parallel in these three environments, serves as the basis for the comparisons. The strengths and weaknesses of each environment will be discussed from the developer's point of view and from the student user's point of view.

DAL Files, by Chris Haupt. This paper is a presentation from the DAL session and sample code for use with Data Access Language (DAL) in either system 6 or system 7.

From Experience, by David Shayer. This is David Shayer's presentation on how to get published, and the ins and outs of having a publisher. From royalties to payment schedules, David discusses the ups and downs of having a publisher. David is the author of Disk Lock and Public Utilities.

Dynamic Object-Oriented Apps on Mac, by Elizabeth Brennan and Mike Russell of Novell. This paper describes a method for implementing a dynamic-linking, object-oriented application platform on the Macintosh. The implementation described is a common, extensible software platform for running network utilities. Our goal was to design a system that enables developers to modify the existing functionality of a utility, and to leverage the functionality of existing utilities when creating new ones. This design is realized by implementing a system that allows developers to inherit or override the functionality of existing utilities. Because the relationships between utility modules are resolved at runtime, the utility developer need not have access to the system source code, or even the source to its objects' parent classes. By using this system, the developer of the new utility gains speed and simplicity of implementation. More importantly, the end user gains an important increase in interface consistency.

Debugging is Hard, by Greg Marriott of Apple Computer's Blue Meanies. Greg was the chief debugger for System 7. He learned many lessons while debugging the software. Greg's presentation covers what he learned about debugging during the System 7 development process.

MacHack™ Schedule '91 in Microsoft Excel Format. This is the full schedule of the 1991 MacHack conference.

MPW Stand-Alone Libraries, by Allan Foster and David Newman of Software Ventures. These are two developers of Microphone. This is a standalone software library for use with MPW.

Multi-Platform Software Development Using the Model-View-Controller Design Paradigm, by Mark Wittenberg, Novell, Inc. Producing a program for more than one platform (such as Macintosh and Windows)

is difficult and resource expensive, and it is difficult to maintain functional equivalence between the versions. Engineering, quality assurance, documentation, and maintenance efforts increase in proportion to the number of supported platforms. Programs with a Graphical User Interface (GUI) make this situation worse. To solve this problem, we present an architecture that reduces the engineering work required to build multiplatform applications, guarantees functional equivalence between platforms, encourages users to form the same conceptual model of the application on all platforms, provides a firm basis for implementing scripting, allows a native GUI for each platform, and simplifies inter-application communication across platforms. The architecture merges the Model-View-Controller and the Client-Server paradigms.

Experiences with Porting Gnu C to MPW, by Stan Shebs of Apple's Advanced Technology Group. As part of its research efforts, Apple's Advanced Technology Group ported the Gnu C compiler GCC to MPW. GCC is a high-quality, retargetable, optimizing compiler that comes with complete and well-documented source code, which makes it an attractive vehicle for experimentation. GCC was originally designed for UNIX systems; the MPW port brings the advantages of GCC to Macintosh developers. Porting GCC to MPW involved several major categories of changes: altering GCC output to be acceptable to the MPW Assembler, modifying the front end to accept Apple's extensions to ANSI C, and altering GCC source code to compile and run under MPW. Because GCC was not originally designed for the Macintosh, some of the adaptations required unusual modifications to the sources, such as storing data inside function name strings. Although the resulting compiler is about three times slower than MPW C and nearly twice the size, its object code is from 5% to 20% smaller and faster than MPW C's output. MPW GCC is mostly compatible with MPW C, and can be dropped in just about anywhere that MPW C is being used.

Getting your products into Magazines, by Steven Howard of *MacWeek.* This is a do's and don'ts presentation for getting editorial placement in magazines.

Setup and Use of Apple's Data Access Language (DAL)—The First Journey, by Christopher Haupt of the Rochester Institute of Technology. Using Apple's Data Access Language (DAL) for the first time can be a confusing journey through different hardware platforms, different

operating systems, various relational database systems, and development environments. This paper provides a brief introduction to the setup and use of DAL from a Macintosh developer's point of view. A brief discussion provides pointers and tips for DAL installation—both on the Macintosh and, in a cursory manner, for the server host. Step-by-step examples follow, which illustrate the application development process; establishing a DAL session, investigating the database environment, manipulating data, and the importance of neatly running down a session are each studied accompanied by C code fragments. The paper concludes with an initial exposure to DAL statements and procedures.

Sheila Brady's presentation is the slide show from Sheila's presentation on the lessons learned during the development of System 7 from the Engineering Manager's point of view.

Testing on a ShoestringTesting on a Shoestring Budget, by Shane D. Looker of the University of Michigan. Testing is the final phase of the product development cycle. Even though testing is done concurrently with coding, it determines when a product is ready to ship. All too often, a product (commercial, shareware, or free) ships with significant bugs that have not been found in the testing process. This is in part due to the number of Macintosh platforms, the complexity of the software, and the differences between versions of the Mac OS. For an individual or small company, it is unlikely that each Macintosh model is available for testing. With careful planning and a directed testing effort, more bugs can be found and fixed before a product's release, while at the same time not requiring all machine types to be used in the testing effort.

Using Object Oriented Languages for Building Non-Applications in MPW, by Allan Foster and David Newman of Software Ventures. Over the past few years, object-oriented programming has been pushed as an appealing approach for Macintosh programming projects. Unfortunately, the tools provided have only allowed this to be used by Applications. This paper shows a technique for using OOP languages for writing standalone code for the Mac OS. Examples of these are INITs, XCMDs, and the various DefProcs for the Mac managers. This paper will show how to use global variables in these code resources, as well as how to provide the necessary framework for both C++ and Object Pascal. Using MPW as the development environment, it will be shown

how a runtime library, combined with a postlink MPW tool, are employed to build object-oriented standalone code. Full support is provided for Object Pascal and C++, including support for C++ static constructors and destructors. It then demonstrates how to build multi-segment code resources, and where they would be used.

Attendee Contributions

In this folder are 19 items that were contributed by the attendees at MacHack. None of them were entered in the Hack contest; each should be used with caution. The material ranges from a technical note on how to deal with apple events in HyperCard 2.0 to a complete GCC C compiler.

1992

Proceedings

Euclid: Supporting Collaborative Argumentation with Hypertext, by Bernard Bernstein of the Department of Computer Science at the University of Colorado at Boulder. Many books and papers are written collaboratively across great distances and over long periods of time. Lately, research has been focusing on real-time collaboration, but little has been done to provide a means to share reasoning like that used in research collaboration. Euclid is a collaborative hypertext system for creating and analyzing reasoned discourse over large spans of distance and time. A Euclid argument may represent a complete argument from a single individual or from many participants with any number of perspectives. As a writing tool, Euclid allows users to fully analyze the logical structure of their arguments to create a clear case when putting it on paper.

The Construction of a TCP/IP to Apple Event Gateway for use in Distributed Computing Experimentation, by Christopher Haupt of the Rochester Institute of Technology. Apple event aware applications open the possibility of sharing unique, application-specific resources

with other Apple event speaking programs. Because Apple events are a System 7 specific feature, it is not possible to use these resources from non-Macintosh platforms without major alterations of the Apple event aware applications. This paper provides a summary of a project to allow non-Mac platforms access to Apple event applications by way of a protocol that facilitates the transfer of pseudo-Apple events over TCP/IP. The TCP/IP to Apple event protocol is discussed, as is an application that implements gateway functionality on the Macintosh platform. To facilitate the use of the gateway, a client API is introduced. This paper documents the first implementation of the protocol and proposes some future enhancements. Two current experiments making use of the gateway are described illustrating possible uses of the protocol.

Colorizing HyperCard, by Tom Hammer, is a set of tools that Tom presented at the 1992 MacHack. The material presented here has been incorporated into HyperCard 2.2.

In the David Shayer, there folder are two talks by David and his wife Ellen Soloman. The first talk was given by Ellen on Copyrights and Trademarks. This talk dealt with protecting software and the names of the products from the competition. The second talk is on getting published. It deals with the real-world product marketing problems.

QuickTime Answers to Random Questions, by Dick Trismen, is a presentation to the attendees at MacHack. Dick was the lead programmer at LetraSet on MediaBlender just before he gave this talk. He was one of the first people outside Apple to be given QuickTime to work with. The talk deals with a number of real issues that QuickTime programmer must deal with.

Experiences from the Development of Harvest C, by Eric W. Sink. This paper describes the development of Harvest C, a full C compiler and linker for the Macintosh. This project has revealed a number of interesting issues, partially due to its unusual size, and partially due to the internal structure of Macintosh applications. The following areas will be considered:

- Memory management for large abstract data structures
- Compiling the usual Macintosh extensions to C
- Macintosh application structure and linking

- The history of Harvest C and its future directions
- Comparisons with commercial C compilers

Harvest C is freely distributable and included in the 1992 Attendee Contributions folder on the CD.

Essence of the Hack are the notes from the presentation by Scott Boyd and Greg Marriott, which covers how to hack and why hacking is fun.

Graphics Acceleration Samples, by Paul Campbell, is from his lecture of the same name. Paul is the genius behind the Thunder Card from SuperMac. Included here are the graphics excelleration routines that he presented as examples of ways to make the screen update quicker.

Macintosh Common LISP, by Michael S. Engber of The Institute for the Learning Sciences at Northwestern University. Macintosh Common LISP (MCL) is a powerful development environment that is often overlooked by Macintosh programmers. This paper will show why you should consider using MCL and will explore some of the ways it can enhance your productivity. Prior LISP or MCL experience is not assumed. This paper and his hacks won Mike a job at Apple where he is now one of the top people in the Newton program.

Optimizing Source Code, by Shane Looker of the University of Michigan. This is Shane's look at making programs go faster by optimizing source code. A large number of real-world hints and tricks are included to make any Macintosh application run faster.

PatchWorks—High-Level Low-Level: Object-Oriented Patching, by Patrick C. Beard. This document describes PatchWorks, an object-oriented, trap-patching system created by Robert (Mouse) Herrell and Patrick Beard, which simplifies trap patching by handling many of the mundane chores that all trap patches must perform. The details of how patches are installed and the assembly language "glue" required to implement patches are taken care of, leaving the programmer free to decide what behavior a patch should perform. PatchWorks supports more than simple trap patching; it supports other patching mechanisms, such as interrupt and exception vector patching, and special purpose mechanisms for debugging and exceptional situations.

PatchWorks also speeds up the development cycle by providing a way for trap patches to be developed as applications, which removes the necessity of rebooting while developing patches. The process of developing operating system extensions is greatly enhanced with the use of PatchWorks. Now the benefits of object-oriented programming can now be used by systems level programmers. While its implementation is specific to the Macintosh operating system, PatchWorks is written in portable C++ and could be ported easily to other platforms that provide some kind of patching mechanism. Robert and Patrick were scheduled to present this paper together, but Robert died. Patrick dedicated the paper to Robert.

Putting an Object-Oriented Face on a Toolbox Manager—How to Use OOP to Manage Toolbox Complexity, by Ralph Krug, P.E. Most Macintosh Toolbox managers are complex units of code. To successfully use a Toolbox Manager, a programmer has to wade through a lot of documentation (Inside Macintosh, Tech Notes, magazine articles) and perform empirical tests. The details of how to use a Toolbox Manager are spread out in an ocean of words and a void of blind tests. Object-oriented programming (OOP) is supposed to provide techniques to encapsulate this kind of complexity.

Richard Clark of Developer University presented a session on AppleScript, which session was broken into two major parts: (1) an overview of the language and tools and (2) a how-to on adding scripting support to an existing application. This slide set is over 70 slides long and contains most of the details of Richard's presentation.

Stephan Somogyi of *MacUser* presented a self-running slide show on how to get five mice in *MacUser* or as it is better known: Rodent Accumulation Techniques. Only the humor is lost from the presentation.

Cognits: A Portable Library of Intelligent Classes, by Dr. Steven M. Lewis, Ph.D. of The Aerospace Corporation. Cognits is a C++ class library designed to facilitate the development of large object-oriented applications. The library includes classes that parallel many of the functions available with MacApp, and other classes are available for scientific graphics and image processing. Unlike MacApp, however, with Cognits, both data and screen objects are built from the same object classes, allowing the use of a model-controller-view paradigm. In this schema,

the responsibility for the construction and maintenance of displays is given to the display objects, rather than to the represented objects. The library consists of approximately 80,000 lines of code defining about 200 classes. Approximately two-thirds of the classes are widgets that are responsible for displaying specific data on the screen. The use of controls and dialog boxes is generally replaced by the use of objects that are responsible for specific sections of a window. Cognits code is written to run in Macintosh, X Windows, and MS Windows environments. Portability is accomplished by generating a virtual graphic interface layer for each target machine. Events on each platform are read, transformed into canonical events on the virtual machine, and passed to system-independent software. Drawing and windowing calls are converted from calls to the virtual machine to calls on the underlying hardware.

Pass the Torch: Enabling the Next Generation of Hackers, by Tom Pittman. Hacking, though intrinsically value free, involves an ethical dimension. A good hack expresses creativity, simplicity, and purpose; a bad hack may be unoriginal or corrupt the system. The main difference between a good hack and an excellent commercial product lies in its size and price; it is quantitative, not qualitative. Furthermore, although hacking is essentially a solo activity, it has important social consequences. Nobody takes up the hacker's hatchet unaided. In some cases, it is another hacker who personally inspires the young initiate; in other cases, it is access to appropriate tools and good sample code. Either way, ultimately, it was a person who did it. Hacking is a valuable asset to society. We repay our obligation to our mentors by inspiring the next generation of hackers through good hacks and personal encouragement.

Macintosh and Windows 3.0: A Developer's Perspective, by Waldemar Horwat. The Macintosh Toolbox and Microsoft Windows 3.0/3.1 are both powerful graphical environments for writing applications. Although many programming details of these two environments are almost identical, they follow quite different philosophies and differ in fundamental aspects. Neither is clearly superior, but their weak and strong points are very different. This paper examines and compares both operating systems from a developer's perspective, highlighting their differences in philosophy. It also contains an appendix that defines object-oriented programming and abstraction and shows one

instance in which they conflict. This may be one of the operating system/application interface conflict areas in the near future.

Attendee Contributions

The 1992 attendee contributions include Disinfectant 2.9 and NewsWatcher from John Norstad. Included with each is sample source code from John's programming libraries and information from his talks. Eric Sink's Harvest C and a self-portrait of the Blue Meanies completes the collection of material from 1992.

1993

Proceedings

Comparing C-Based Object Systems, by Gary Odom of Electron Mining, is an overview of the important issues of object-oriented programming and a comparison of C-based object systems (Objective-C, C++, and OOPC). This paper compares C-based object systems. To provide a basis for that comparison, the articles begin with a perspective about why OOP is important, what the important issues are with object orientation, and a brief mention of other object systems from which designers of C-based object systems have drawn their inspiration.

Cognits: Writing Portable Code, by Dr. Steven M. Lewis, Ph.D. of The Aerospace Corporation. Writing good, user friendly, graphical applications on the Macintosh is difficult. Even more challenging is writing a single application capable of running across a number of GUIs. This paper is a case study of the Cognits library, a portable class library that runs on the Macintosh, X Windows, and MS Windows. The discussion will specifically concentrate on writing code that will port between the Macintosh and MS Windows. Three areas will be addressed: (1) the general principles in the design of portable systems; (2) corresponding features of the Mac and Windows operating systems, especially drawing and interaction with the user; and (3) unification of higher-level functionality. The object of a portable library is to allow a

single collection of source code to compile into applications that will run under multiple systems. While higher-level code may manipulate system-dependent structures, well-designed portable systems should not require changes in the text of code to generate similar results. On multiple systems Cognits divides code into a system-dependent layer and a larger system-independent layer. The system-dependent layer interacts with the underlying GUI and defines a ''virtual machine' used by high-level, system-independent code. The most important step in designing portable code is to unify disparate conceptual views of elements of the applications into a single uniform framework. Correspondence must be made between elements of the two systems, leading to a single integrated approach to drawing, event handling, and controls. Important decisions must be made concerning implementation of controls, whether to use native or portable look and feel, and where events are handled in an application. Finally, there are differences in the function of advanced features such as standards dialogs and interprocess communication. This paper considers a number of choices in implementing Cognits and presents code to implement many of the common Macintosh drawing commands under Windows.

Communication Abstractions in Concurrent Processing, by Waldemar Horwat of MIT. Parallel and distributed programs are entering the mainstream—a 1024-processor Connection Machine is now the fastest computer in the world, while at other places, users are routinely running distributed programs on workstation networks. One of the most important aspects of parallel programming is the abstraction of the communication channels between processors as presented to the programmer—how is communication implemented among processes in a program in such a way that is very efficient both when the processes are on the same node and when they are on remote nodes? This paper presents and compares several approaches of viewing communication, including cache-coherent shared memory, message passing, and higher-level protocols. The important hardware constraints are examined, some predictions for the future are presented, and a number of literature references are provided.

Stephan Somogyi of *MacUser* made a presentation on Data Security. The "slides" are part of a self-running app generated by Peirce Software's Smoothie. You will need to have QuickTime (any version)

running for the slide show to work since it uses QuickTime's compression for the slides. The presentation covered Clipper, RSA, PGP, and other encryption methods.

David Feldman of Highland Capital Partners made a presentation on Venture Capital. This presentation was well attended, and David's talk covered both sides of the game. At Infini-D, he helped land venture capital from Highland. At Highland, he is identifying the next generation of high-tech companies. The slides include the PowerPoint viewer, which you must open before you open the slides from within the viewer.

Better Development Tools Are Coming, Will They Be Good Enough? by John Clark. It's becoming increasingly apparent that there is much need for improvement in software development tools. New and improved development environments are coming out for the Macintosh almost monthly. This paper discusses known techniques for making software easier to develop and reviews the major products coming to market. It also reviews and amplifies the reasons we need better tools and the costs of developing with poor-quality tools. Existing strategies offer partial solutions to the problem. A proposal is made for an alternative approach that allows for not only known techniques, but also expected future techniques. These can be incorporated into a single system that does not require starting over from scratch when techniques are added.

Echo Logic Slides is available from the session that was given by Echo Logic on how to use the Echo Logic software to beat the power curve on getting software ready for the PowerPC.

Making Your Application AOCE Savvy, by Steve Falkenberg. This PowerPoint presentation has all the information required to retrofit an application that already exists with AOCE calls. The slide set has complete details in it.

Images of Ourselves: the Electronic Little People, by Murita Poulff of Digital Equipment Corporation. Interaction with computers is changing. Increasing capabilities and complexity has led to a controversy over the appearance of user agents: Should they or shouldn't they be made accessible through anthropomorphism? Those who say "yes," believe it makes agents' action more understandable, and that this predictability helps users direct them more effectively. Those who disagree believe that the "human-like helper" metaphor distorts rather than clarifies.

This paper shows that the two views are not incompatible and shows some implications for developers. Three examples of agent-based systems are briefly examined.

Listen to Your Hack Beat, by Bernard Bernstein and Chris DiGiano of the University of Colorado. When writing code in an event-oriented environment, many things may happen at any given time. Debugging environments provide the programmer feedback about the execution of the code, but the information is often limited to textual snapshots of the program state. With the use of audio debugging points, we can "listen" to the code as it executes, and we can hear when something wrong happens. This paper describes what a sonic debugger might sound like and how it would work.

The Macintosh as an Internet Server, by David Peterson. The Macintosh is appearing more and more frequently on networks alongside UNIX hosts. Users and network managers alike expect these Macs to provide the network services that are common to these other machines. The Internet Server program itself is written as a faceless background application. It performs no idle processing and as such consumes CPU time only when it must launch a program to service a remote machine. When this occurs, the server looks up the appropriate program and launches it with the MacTCP stream pointer as the program's launch parameters. This makes the first event received by the program contain the stream pointer for the connection it must manage. There is also a mechanism through which the launched program can request that asynchronous notification events related to the stream be sent to it. This approach gives the program that implements the service complete control over the IP connection just as if it had actually been created by the program. It also separates the listening for service requests from the actual handling of the service, yielding advantages. The Internet Server is managed by a Control Panel that allows users to specify which TCP and UDP ports to listen on, and to choose what program to launch when a request is received. The configuration process is presented in a familiar Mac-like way—there are no text files to edit or formats to remember.

The whole system is written as a series of programs with a Control Panel, and there are no drivers or system patches to install that might cause system conflicts. It is easily expanded to provide other network

services, and it enables the Macintosh to provide common services to other machines on the network. A full source is provided on the CD.

Parallel Processing on a Macintosh Network, by Shane D. Looker of the University of Michigan. Parallel processing is a powerful method of solving some types of computational problems. Complex problems (such as graphic rendering) can take hours to complete, even on the fastest Macintosh. If a problem can be broken down and solved in parallel, the solution time can potentially be cut in direct proportion to the number of processes that are dedicated to the problem. Several microprocessors have parallel processing integrated into them, but the current generation of Macintoshes do not use these chips. Utilizing System 7, a network of Macintoshes can work as a distributed parallel-processing engine to solve complex problems in less time than a single machine. Methods of target processor selection, task distribution, remote procedure calls, and process synchronization are discussed.

Assembly Language Programming and Optimization Techniques for the POWER Architecture, by Gary J Kacmarcik of Case Western Reserve University. This paper presents an overview of the POWER architecture and discusses techniques for writing efficient assembly language programs for machines based on this architecture (and its derivatives). IBM's RS/6000 is currently the most commonly available machine that is based on this architecture, and thus, it is the machine that is used as the focus of this presentation. After a brief description of the architecture and the available instructions, optimization techniques, which are specific to POWER programs, are presented with a discussion of why they are important. In addition, a set of tools for the Macintosh, which allows programmers to assemble, execute, and analyze POWER assembly language programs, is described. These tools operate on standard RS/6000 XCOFF format .o and .obj files. Immediately after finishing this paper, Motorola released the technical information for the PowerPC 601. While time constraints prevented incorporating more detailed timing information about the 601, the instruction set summary in Appendix A has been changed to incorporate the new information.

PowerPC A & B, by Jordan Mattson and Richard Clark of Apple Computer. These are the slides from the PowerPC sessions at MacHack.

Richard and Jordan have outlined in great detail the programming steps needed to create PowerPC native applications.

Shapiro/QuickTime Talk outlines Eric Shapiro's presentation on QuickTime. Eric built the talk from experience. He is the author of Video Beep and Spectator, both of which use QuickTime.

Swedish to You is like Greek to Me, by Jon Wätte. Reaching a global audience is good, both for the ego and the wallet. However, serving customers properly outside a home country requires some thought. Here is a summary of some of the more important issues. The focus is on how to resolve these issues in applications running under the Mac OS.

Tektronix Patent Info is a folder filled with the information on the ill-fated digital video patent that Tektronix tried to enforce.

Making Mac Listen: A Voice Recognition Toolkit for Macintosh Applications, by Alma Whitten and Robert McCartney of the University of Connecticut. Commercial products now exist for the Macintosh, which can perform recognition of discrete utterances for a set of pre-trained words. The question arises as to how this capability might be integrated into and used within an application; in particular, how we might integrate such capabilities into an application without radical redesign, while maintaining its original nonvoice capabilities and appearance to the user. They have developed and implemented a toolkit in Macintosh Common Lisp, which can be used with any voice recognition product capable of generating an AppleEvent with a recognized utterance as a string parameter. The toolkit is a package consisting of centralized processing code and a set of specialized versions of standard MCL user-interface objects, such as windows, buttons, and other dialog items. Integrating the toolkit into an application allows the user to refer to any on-screen object by a sufficient subset of its text label, causing the object to respond as if it had been mouse-clicked. All such processing is transparent to the application designer, who need to merely substitute the provided object types for the standard versions and to include the processing code. Thus, a level of voice recognition capability is thus provided. This capability dynamically responsive to the state of the screen display, which requires no pretraining beyond the association of a sufficient vocabulary of discrete words with AppleEvents, and which allows the user to mix voice input with

mouse and keyboard at any time. In this paper, they compare their approach to the alternative of predefining an action for each utterance to be recognized. They discuss the algorithms, specialized user interface objects, and data structures used to maintain information about the screen contents and to support incremental recognition of on-screen objects. The hardware/software product currently being used to recognize individual words is the Voice Navigator II SW from Articulate Systems, Inc.

Attendee Contributions

Crack me!, by the DTS staff at Apple is a puzzle for programmers to crack. The contest is over, but read the directions and have at it.

GCC-2.3.3r9, by Stan Shebs, is a port of the GNU compiler to the Macintosh. Stan did some wild things with this compiler at MacHack.

Listen to your hack...beat, by Bernie Bernstein, was created entirely at MacHack and is based on the paper covering audio debugging tools.

Splash is a very small Symantec C project.

Voice Toolkit is the source code in Macintosh Common Lisp that goes with the paper of the same name.

Xconq 7.0d15 is a port of the UNIX game system by Stan Shebs. This is the code that he so diligently poured through the GCC compiler during MacHack. The game is not fully debugged, but it runs fairly well.

Index

C

D

E

F

G

H

I

J

K

L

Q

R

S

T

U

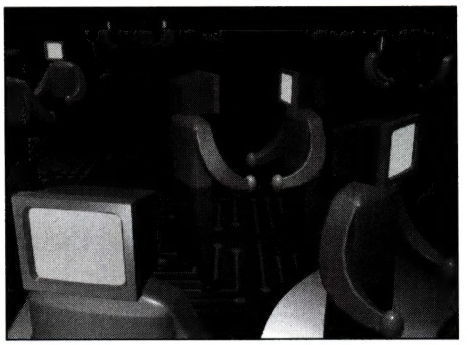
V

W

X

Y

Z



Afterword

Eight-Year Veterans of MacHack

Amieé Moran
Doug Houseman
Leonard Rosenthal
Jay Newman
Bill Johnson

For More Information on Future MacHack Conferences Contact:

Expotech
1264 Bedford Rd
Grosse Pointe Park, MI 48230-1116
313-882-1824
Applelink: Expotech
AOL: Expotech
Internet: Expotech@aol.com

How to Use the CD-ROM

The CD is fairly straightforward. It is set up like a volume on a hard disk, with each top-level folder refering to a year at MacHack. You will find two things in that folder. The first is a HyperCard stack that has information about all the hacks in it. Each card in the stack is written by the author of the hack at the time the hack was entered in the hack contest. The second thing you will find are the hacks themselves. In the hack folders, will be the hack, any documentation that was submitted by the programmer, and source code, if it is available.

To use the hacks, place the CD in your CD-ROM drive and it should appear on the desktop. Open the folder for the year that contains the hack you are interested in and then open the folder with the hack in it. If it is an executable program, you will be able to double-click it and run it. If it is an INIT or extension, then you will need to place it in your system folder and restart your machine. Once you have done that, the hack should be active and available to you.

Do not try to run more than one hack at a time, unless the documentation recommends it. If one hack can make your machine a little flaky, two can make it unstable, and three can make it unuseable. Each hack was tested, before being put on the CD, to make sure that it would run, but none of them have had the level of testing required for commercial software.

NOTE

If you have a problem with booting once a hack is installed, boot from a floppy and remove the hack in question, if you are running System 6. If you are running System 7, reboot and hold down the **Shift** key to turn all of your extensions off.

WARNING

It is recommended that you remove all the other extensions in your system folder when you are experimenting with the hacks. You can do this quickly by creating a "not in use" folder and dragging all the other extensions or INITs into it. This eliminates many conflicts. If you really want to use a hack on a daily basis, test it carefully, before you decide whether it is going to do the job for you. This is a word to the wise, not only for the INITs and extensions contained on this CD-ROM, but for all the INITs and extensions that you might want to run.

LATE NIGHT WITH MAC HACK

Each year, 300 of the top Macintosh programmers get together at the renowned MacHack conference, where they compete for the outstanding prize of the Best Hack award. Over 100 hacks have been created during the last few years, and most of them have been unavailable... until now! *Late Night with MacHack* allows you to meet the programmers, try their hacks, and follow the vision of the Mac interface from the early years to the present day.

MacHack gave way to color on the desktop, screensavers, desktop printing and services, icons, and more. And Apple has recruited many programmers from MacHack. This book tells the story of how the conference has become an urban legend that others have tried to duplicate with no success. Here is the story of MacHack, the Hackers, and their invaluable Hacks—System 7, Disklock, and TCP/IP for the Mac, to mention a few.

Doug Houseman is the Program Chairperson of MacHack, as well as one of its founding members. He is a regular speaker at MacWorld, and other Mac technology conferences, and has written about the Macintosh industry for the last 8 years. His articles have appeared in *MacTech Quarterly, MacTutor, MacWorld Magazine*, and numerous foreign and user group publications.

WHAT YOU WILL FIND ON THE CD:
Over 100 of the best hacks created at MacHack, including:
The Grouch • NetBunny • Jurassicon Park • DropSave • QuickTime Balloon Help • The Mac Clapper • Wavy •
and more...

- **Covers over 100 of the Hacks from MacHack—including NetBunny**
- **A look at the way System 7 was shaped**
- **Where the programmers are driving the next generation of Macs**
- **Experiment with prototypes of unreleased Apple system software features**
- **Learn how programs are created in the hot house of MacHack**
- **Test your own programming skills against the best Mac programmers**
- **Use the award winning programs from the Best Hack contest, provided on the CD-ROM**

ISBN 1-55851-395-7
US $29.95
CAN $41.00
90000>
9 781558 513952

M&T BOOKS

LEVEL
Intermediate
Programming
Macintosh